Darmstadt University of Applied Sciences

-Electrical Engineering and Information Technology International-

-Master of Science-

IMPLEMENTATION OF AUTONOMOUS MOBILE ROBOT IN A TRADITIONAL WAREHOUSE MANAGEMENT SYSTEM

Rutvik Prakash Pradhan - 1119301

Aayush Kaushik - 1119305

Touhid Ahmed - 1112281

Prakash Ganesan - 1119221

Bhavinkumar Prajapati - 1119282

Rajbansi Devmani Kamalbahadur - 1119291

Project Supervisors:

Prof. Dr. Ing. Sven Rogalski
Mr. Leon Pfenning

Abstract

Conventional Warehouse management systems are dependent on humans for the completion of warehouse tasks such as inventory management, transporting goods, etc. which consumes a lot of time, leading to poor productivity and can even lead to safety and health concerns for the people working there. To tackle such problems and keep up with the competition in terms of productivity levels, Autonomous mobile robots (AMR) are currently being introduced in many intralogistics operations in the warehouses. Their advanced hardware and control software allow autonomous operations in dynamic environments. Compared to an automated guided vehicle (AGV) system in which a central unit takes control of scheduling, routing, and dispatching decisions for all AGVs, AMRs can communicate and negotiate independently with other resources like machines and systems and thus decentralise the decision-making process. Decentralised decision-making allows the system to react dynamically to changes in the system state and environment.

This identifies a real-world problem of a conventional warehouse management system, that has a lot of features, such as shelves and introduces an Autonomous Mobile Robot named TC200, that autonomously moves within the warehouse workspace while avoiding dynamic and static obstacles and reaches the shelf location. Then it moves holonomically about a feature to reduce trajectory time and which uses a fine-tuned robust computer vision algorithm to detect different coloured objects (yellow and red storage bins in this case) within the warehouse, via a camera mounted on the TC200 AMR. The main aim of TC200 is to move autonomously within the given warehouse and detect the bins and provide some statistical information regarding the objects to the user. The future scope of this project involves a manipulator which can be mounted on the robot, to collect these detected bins, and then to move them from one place to another, thereby decreasing the transportation time of objects and increasing the overall productivity of the warehouse.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

1.1 Background

Warehouses are large buildings used by manufacturing companies, importers, exporters, wholesalers, and many other businesses where the employees receive, sort, stock, and prepare goods for distribution. The two types of warehouses are Traditional/Conventional Warehouses and Smart/Modern warehouses.

1.1.1 Traditional/Conventional Warehouses

Traditional warehouses have long been the cornerstone of supply chain operations, serving as storage facilities for goods before distribution. These warehouses are typically characterised by manual processes and labour-intensive operations. Warehouse workers are responsible for tasks such as receiving, inventory management, order picking, packing, and shipping. While traditional warehouses have served businesses for many years, they come with inherent limitations and challenges.

Challenges faced by Traditional Warehouses

- **Full Dependency on Manual Labor**

  One of the primary challenges of traditional warehouses is the reliance on manual labour. Finding and retaining enough human workers to meet the demands of the warehouse is very hard in industries where the work can be physically demanding and less appealing, and this leads to very high labour costs. Human involvement in various tasks increases the likelihood of errors and inefficiencies.

- **Decreased Productivity**

  Order fulfilment in traditional warehouses can be time-consuming and prone to errors. Warehouse workers manually pick items from shelves, pack them, and prepare them for shipping. These manual processes are susceptible to mistakes, leading to delays in order processing and potential customer dissatisfaction. Moreover, the lack of optimised routing and warehouse layout can cause inefficient movement within the warehouse, resulting in wasted time and effort.

- **Inventory Management**:

  Manual inventory management processes, such as manually recording item counts and tracking stock movement, are prone to inaccuracies and can result in discrepancies between actual inventory and recorded stock levels. This can result in stockouts, where products are out of stock when customers want to purchase them, or overstocking, where excess inventory ties up capital and increases carrying costs. Manual processes can be slower and less efficient than automated ones. This can lead to longer lead times, slower order processing, and higher operational costs.

- **Scalability**

  Traditional warehouses may face challenges in adapting to changing business needs and scaling operations. The infrastructure and layout of traditional warehouses are typically rigid, making it difficult to accommodate growth or changes in product assortments. Expanding storage capacity or introducing new processes may require significant time and resources.

1.1.2 Evolution of Smart/Modern Warehouses

The meteoric rise of eCommerce and the popularity of online platforms like Amazon, the supply chains are challenged to deliver products faster while keeping customer costs low, all while offering same-day or next-day delivery. This fulfilment is not possible with traditional warehouse management systems. So, smart warehouses are evolving more than ever.

Smart warehouses integrate a range of technologies to streamline operations and enable intelligent decision-making. Automation plays a pivotal role, with robotics and machinery taking over repetitive and labour-intensive tasks. This automation reduces reliance on manual labour and enhances the speed and precision of warehouse operations.

An AMR, or Autonomous Mobile Robot, is typically used as part of this automation. It is a type of robot designed to operate and move autonomously without human intervention. These robots are equipped with sensors, cameras, and advanced navigation systems that allow them to perceive their environment, make decisions, and navigate through spaces safely and efficiently. These robots are advanced enough that they can work together with humans to make operations better and more efficient. They can do things like carry heavy stuff and travel long distances which humans are not capable of doing for extended periods of time, which allows humans to do other important tasks.

Benefits of Smart Warehouses over Traditional Warehouses

- **Increased Efficiency**

  Smart warehouses optimise order fulfilment and contribute to increased productivity through automation. Robotic systems and AMRs automate the picking, packing, and sorting operations, significantly reducing processing time and errors. Optimised routing algorithms and advanced warehouse layouts further enhance efficiency, ensuring that orders are picked, packed, and shipped with precision and speed.

- **Inventory Management**

  Smart warehouses revolutionise inventory management through real-time tracking and visibility. Automated systems, such as barcode scanners and RFID tags, enable accurate and efficient inventory control. With instant updates on stock levels and locations, businesses can make informed decisions to prevent stockouts or overstocking, improving customer satisfaction and reducing carrying costs.

- **Scalability and Flexibility**

  Smart warehouses offer greater scalability and flexibility compared to traditional warehouses. Modular designs allow for easy expansion or reconfiguration to accommodate changing business needs. These warehouses can seamlessly integrate with other systems, such as ERP or TMS, enabling end-to-end visibility and smooth data exchange across the supply chain.

- **Increased Safety**

  Smart warehouses prioritise worker safety and ergonomics. Collaborative robots (co-bots) work alongside human workers, reducing physical strain and improving safety. AI-powered safety systems can detect and prevent potential accidents. Overall, smart warehouses create safer working environments, improving employee well-being and productivity.

Structure of the Report

The report starts with an Abstract that offers a brief but comprehensive glimpse into the project's scope, significance, and outcomes. The table of contents is then listed, which provides a detailed sectional view of the topics that are going to be covered in this report. After that, comes the Chapter 1 - Introduction, which categorises warehouses into Traditional and Modern, and then states the major problems faced by Traditional and how the Modern one solves those problems. Then, the problem statement is stated. Chapter 2 is the Literature Review that highlights and showcases the similar solutions to the problem statement that are currently being implemented in the real world. Taking inspiration from the Literature review, a solution was proposed for the problem statement by providing a detailed description of the AMR that was chosen for the specific use case, i.e., TC200, taking into consideration its locomotion and path planning techniques. Chapter 3 starts with the use case overview and then talks about the concepts behind the implementation of the proposed solution by dividing the solution into two main features of Autonomous Navigation and Object Tracking and then stating the system architecture and the algorithms behind both the features. Further in Chapter 3, the methodology of the Computer Vision Algorithms that are used for robot perception, which include the pipeline of the Deep Learning Model, Dataset generation, Model training and finally the model deployment on TC200 is stated. Coming to Chapter 4, find the implementation of the modules that were described in Chapter 3 for both the Autonomous Navigation and Object Tracking features are described. Once the implementation was successful, the validation of the implemented modules for both the features was the next step, which is explained and showcased with real data, in Chapter 5. This validation involved the use case validation, the validation of the Object Detection and Instance Segmentation Model, the validation of Auto Annotation, the validation of Deployment and the validation of laser scan data of TC200. Once the validation was approved, the end goal of the project was reached and all the content described in Chapters 1-5, was concluded at the end of the report, along with the Citations and Appendix.

1.2 Problem Statement for the Project

Given is a Traditional Warehouse Setup with a lot of features like shelves, boxes, garbage bins, etc. The shelves contain different coloured storage bins as well as other objects and the warehouse task involves the detection of certain objects (red- or yellow-coloured bins in this use case), in the whole warehouse environment.

The current warehouse management system depends largely on manual labour for the required warehouse tasks including the sorting, segmentation and transportation of these colour-coded storage bins. The worker would move along all the shelves in the warehouse and would look for the desired coloured bins. On detecting the required bins, the worker would then collect them and transport them to the desired location. Due to large dependency on manual labour, this process is very time-consuming, inefficient and can lead to large human errors, thereby resulting in lower productivity.

Therefore, the main aim of the project is to increase the overall productivity of the warehouse by decreasing process times, decreasing the large dependency on manual labour and decreasing the errors that were very prominent in the case of manual task management.

Chapter 2: Literature Review

2.1 Current Implementation of AGVs and AMRs in Warehouses

In [1] the author talks about the work that was carried out in an automotive company. The main objective was to increase the productivity of the last workstation of an assembly line by implementing an AGV System to transport finished goods to the warehouse. The ultimate goal was the elimination of human-operated transport vehicles like forklifts from the production area, boosting not only flexibility but also workplace safety. He talks about the incorporation of Lean Manufacturing to eliminate waste and meet customer demands for innovative, cost-effective products and the implementation of AGVs to automate material handling, optimise processes and reduce waste. According to the author, implementing AGV systems enhances efficiency and aligns with evolving customer expectations. The implementation of AGV technology yielded significant benefits. It eliminated the manual transport of heavy loads by operators and reduced excess containers at workstations, thereby improving ergonomics, safety, and productivity. Workers no longer needed to handle containers 100 to 150 times per shift. Over a full shift, this translated to (i) eliminating the manual transport of 36 to 39 tons of material by workers, (ii) reducing worker walking distance by 1.75 to 2.6 kilometers, and (iii) saving 50 minutes of travel time. With enhanced safety features, accidents between personnel and AGVs were less likely, contributing to overall workplace safety. Additionally, the risk of damaged components during transport was minimised. The author in [2] highlights the advancements in technology that came with the introduction of AMRs and how they are a huge step forward over the current implemented AGVs. The research done by the author in [2] explains that Autonomous Mobile Robots (AMRs) have evolved to meet the growing demand for flexibility in various industries. Unlike Automated Guided Vehicles (AGVs), which are limited to repetitive tasks, AMRs excel not only in navigation but also in offering diverse services. They can handle tasks beyond material transportation, including patrolling and cooperating with human operators. With the ability to make independent decisions, AMRs provide adaptable solutions. He concludes by stating that AMR technology advances enhance operational flexibility, productivity, quality, and, occasionally, cost-efficiency. Rapid implementation is feasible, especially where suppliers have expertise. However, quantifying AMR benefits and optimising deployment remains challenging. Simultaneously addressing multiple decision variables like vehicle numbers, zoning, service points, scheduling, and path planning enhances understanding and promotes balanced decision-making. Meanwhile, the author in [3], extends the scope of AMRs by integrating Internet of Things (such as RFID, smart sensors, etc.) and Cyber Physical Systems, that connect via the Internet to communicate with each other. According to the author, RFID Technology is one of the most used IoT technologies in the warehouses. However, he states that Implementing advanced automation systems like warehouse robots with IoT technologies demands substantial investments and meticulous strategic and tactical planning. The future holds the promise of widespread adoption of highly automated systems that will fundamentally transform warehouse operations. Given the evolving warehousing industry landscape, research focused on upgrading existing warehouse systems should receive increased attention and consideration. Also, the author highlights the importance of Warehouse Sustainability by considering factors like power management, the design and operation of the logistics chain and other human factors. He sums it all up by concluding that AMRs and smart technological advancements are the next steps for any traditional or semi-modern warehouse in order to keep up with the rapid order fulfilment

and customer demands, due to the meteoric rise of Ecommerce and competition from other industries.

In conclusion, after going through the various publications, stating the current implementations of Robot Technologies such as AGVs and AMRs, it can be said that the introduction of such robot technologies have a significant boost on the overall productivity of a warehouse when compared to a traditional warehouse, that is completely dependent on Manual labour for the completion of warehouse tasks. AGVs and AMRs benefit traditional warehouses by automating tasks, reducing manual labour, enhancing safety, and boosting efficiency. AGVs handle repetitive tasks, while AMRs offer flexibility and versatility. Both technologies help warehouses adapt to modern demands, ensuring competitiveness, faster order fulfilment, and sustainability. The integration of Cyber-Physical Systems and IoT Systems such as RFID Scanners, enhance the use case and scope of AGVs and AMRs in warehouses, even more.

Therefore, most of the world has moved to a modern warehouse, from a traditional warehouse due to the competitiveness, sustainability and increased productivity that it offers, and this is the way to go, if an Industry wants to survive in this competitive environment. The project draws motivation from the facts stated above and intends to implement a solution by taking into account the capabilities of AMRs in the context of warehouse management.

2.2 Computer Vision in Industrial Automation

Computer vision is a field of computer science that deals with the extraction of information from digital images or videos. It is a type of artificial intelligence (AI) that allows computers to "see" and understand the world around them. It is a rapidly growing field with applications in a wide variety of areas, including automation, robotics, medical imaging, and self-driving cars.

2.2.1 Traditional vs Deep Learning method of Computer Vision

i. Traditional Methods of using CV in Industrial Automation:

> Historically, computer vision in industrial automation relied on conventional techniques like image processing, feature extraction, and rule-based algorithms. These methods often required extensive manual engineering, making them labour-intensive and limited in their adaptability to complex environments. For instance, in quality control processes, traditional computer vision systems employ techniques like edge detection and colour-based segmentation to identify defects in manufactured goods. While effective in certain controlled settings, these methods struggled with variability in lighting conditions, object orientations, and occlusions. As a result, their application was restricted to specific use cases within industrial automation.

ii. Deep Learning-based Computer Vision for Industrial Application:

> The emergence of deep learning has revolutionised the field of computer vision, particularly in industrial automation. Deep learning models, specifically Convolutional Neural Networks (CNNs), have demonstrated exceptional capabilities in tasks such as

object recognition, segmentation, and anomaly detection. In industrial contexts, CNNs have been leveraged for tasks like automated inspection, where they excel in learning complex patterns and features directly from raw image data. For example, in manufacturing, CNNs have been deployed to inspect products for defects, achieving higher accuracy rates compared to traditional methods. Moreover, the versatility of deep learning allows for transfer learning, enabling models trained on large datasets to be fine-tuned for specific industrial applications. This adaptability has significantly expanded the scope of computer vision in industrial automation.

2.2.2 Potential of Deep Learning in Warehouse Automation

In an Industrial set-up, traditional methods of Computer vision are still highly relevant in many cases [4], such as when there is a small amount of training data available or when the task is computationally expensive. However, the implementation of deep learning models in industries is increasing exponentially [5]. This is especially true in warehouse settings, where the practical applications of deep learning-based computer vision are vast [6].

- **Object Detection and Tracking**

  Object Detection identifies objects in an image or video frame, labelling them with a class and bounding them with a box. It involves following these objects over consecutive frames, maintaining their identity and predicting their future positions. This technology is valuable in warehouses for locating and monitoring products, pallets, and robots, leading to enhanced efficiency in tasks like order picking and inventory management.

- **Scene Understanding**

  This can be used to understand the layout of a warehouse and the relationships between objects. This information can be used to plan the movement of robots and other equipment and to optimise the use of space in a warehouse.

- **Quality Control**

  This can be used to inspect products for defects or damage. This information can be used to improve the quality of products and to reduce the number of returns.

- **Inventory Management**

  It can be used to track the inventory levels of products in a warehouse. This information can be used to ensure that the right products are in the right place at the right time.

- **Safety Monitoring**

  This can be used to monitor the activities of workers and robots in a warehouse to prevent accidents.

Different Deep Learning Models

When it comes to selecting the right Deep Learning model for object identification in Computer Vision (CV), a few key aspects come into play.

Firstly, the design of the model, known as its architecture, holds great importance. Models like Convolutional Neural Networks (CNNs) excel at extracting details from images, making them

highly valuable for CV tasks. It's also crucial to strike a balance between speed and accuracy, especially in fast-paced scenarios like robotics or augmented reality. Additionally, if your task involves pinpointing specific areas or objects in an image (known as semantic segmentation), a model adept at predicting masks becomes essential. Considering the depth of the network is also important; deeper networks can capture more complex patterns, but they may require more data and computing resources. By weighing all these factors, you can select a model that aligns with the specific needs of your CV application, ensuring optimal performance and accuracy.

Various models used for Object identification with key features comparison is in the Table 1.

Feature	**YOLOv8**	**SSD**	**Faster R-CNN**	**Mask R-CNN**
Architecture	Single-shot detection	Single-shot multi-box detector	Region-based convMask R-CNNolutional neural network	Region-based convolutional neural network with mask branch
Speed vs. Accuracy	Fast with good accuracy	Slower with high accuracy	Slower with high accuracy	Slow with highest accuracy
Real-time Applications	Object detection, instance segmentation	Object detection in high-resolution images	Object detection, instance segmentation	Instance segmentation
Mask Prediction	Available in YOLOv8	Not available	Not available	Available
Number of Stages	Single-stage	Single-stage	Two-stage	Three-stage
Key Focus	Speed and good accuracy	Balanced speed and accuracy	Accuracy and detailed object detection	Accuracy and instance segmentation
Resource Requirements	GPU/CPU	Less Demanding	High-End GPU	High-End GPU
Real-Time Processing	Yes	Yes	Yes (with GPU)	Not Ideal
Community Support	Active	Active	Active	Active

Table 1: Comparison of popular Single and Two Stage Object Detectors

Yolov8 vs other Yolo Versions

In this comparison, one can assess YOLOv8 in contrast to previous YOLO versions, with a focus on aspects such as speed, accuracy, and features.

- YOLOv8 has better accuracy than previous YOLO models.
- The latest YOLOv8 implementation comes with a lot of new features, especially the user-friendly CLI and GitHub repo is noteworthy.
- It supports object detection, instance segmentation, and image classification.
- The community around YOLO is incredible, just search for any edition of the YOLO model and one can find hundreds of tutorials, videos, and articles.
- Training of YOLOv8 will be probably faster than the other two-stage object detection models. [20]

The YOLOv8 model family offers various models based on their size. Here is a comparison of the different models [7] in Fig.1

Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

- **mAP^{val}** values are for single-model single-scale on COCO val2017 dataset. Reproduce by `yolo val detect data=coco.yaml device=0`
- **Speed** averaged over COCO val images using an Amazon EC2 P4d instance. Reproduce by `yolo val detect data=coco128.yaml batch=1 device=0/cpu`

Fig. 1: Comparison of Yolov8 Model based on size

2.2.3 Why Instance Segmentation?

- The difference between instance segmentation and object detection techniques is that object detectors only detect objects in images. Conversely, instance segmentation solutions provide a fine-grained understanding of image data by defining and classifying each instance present in visual input. [21]

- Robust to occlusion: Segmentation is more robust to occlusion than object detection. This is because segmentation does not rely on bounding boxes, which can be easily occluded by other objects in the image. [22]

- Accurate for irregular objects: Segmentation is more accurate for irregular objects than object detection. This is because segmentation does not rely on bounding boxes, which can be difficult to fit around irregular objects. [21]

- Precise Object Localization: Segmentation provides pixel-level accuracy by classifying each pixel in an image as belonging to a specific object class or background. This precise localization is especially useful when you need to precisely outline the boundaries of objects in an image. [23]

2.2.4 Challenges in using Computer Vision

Despite the promising potential of computer vision in warehouse automation, several challenges persist. One significant obstacle is the need for robustness in real-world warehouse conditions. [5] and [6] discusses the key challenges in using Computer Vision algorithms such as:

1. Variability in Environmental Conditions
2. Complex Object Occlusions:
3. Generalisation to New Object Types
4. Real-Time Processing Requirements
5. Adaptability to Dynamic Environments
6. Large-Scale Data Acquisition and Annotation

2.3 Proposed Solution

The major drawback of the given problem statement is with its Warehouse Management System, which is largely dependent on manual labour. To achieve the aim of increasing the overall productivity of this warehouse, the implementation of modern practices and converting this traditional warehouse into a Smart/Modern warehouse was a must.

To do so, a comparison between the currently implemented Automation technologies such as AMRs or AGVs was conducted, and then the best solution to this Traditional Warehouse Management System, which would significantly decrease the task completion time was decided, which would lead to fewer errors and would increase the overall Warehouse productivity. The factors used for comparison between AMRs and AGVs were:

- **Navigation Capabilities**

  AMRs use LiDAR, cameras, and sensors to navigate and adapt to changing environments. They can create maps of their surroundings in real time and plan their routes dynamically whereas AGVs typically rely on fixed infrastructure, such as magnetic tape, wires, or QR codes, for navigation. They follow predefined paths and

require a structured environment to operate efficiently and would fail to function in case of change in the warehouse environment.

- **Flexibility to changing Environment**

  AMRs are highly flexible and adaptable. They can navigate around obstacles, change routes on the fly, and handle dynamic layouts. This flexibility makes them suitable for warehouses with frequently changing inventory or layouts. On the other hand, AGVs are less flexible and require a more structured and predictable environment. Any changes in the warehouse layout may necessitate updates to the physical guidance system.

- **Scalability**

  AMRs are scalable, and you can easily add or remove robots to match changing demand. They can also work collaboratively, making it easier to adapt to varying workloads, whereas, expanding an AGV system may involve additional infrastructure installation and adjustments, which can be less straightforward than adding AMRs. [8]

- **Safety**

  AMRs are equipped with advanced safety features, such as obstacle detection and collision avoidance systems, which make them safer to operate around humans and other equipment. AGVs can be safe when used in controlled environments but may have limitations in terms of safety features compared to AMRs as they don't have an intelligence system and in case of an obstacle, it will wait for help, instead of finding a way out on its own. [9]

Taking inspiration from the Literature Review mentioned above, considering the shortcomings as well as benefits of the currently implemented techniques, and then comparing both AMRs and AGVs, the solution to integrate an AMR with this Traditional Warehouse was chosen because it was required not to restrict the solution to the current warehouse layout and that the solution will be flexible to future changes as well as be scalable.

Objectives of the Proposed Solution

- Navigating autonomously within the warehouse, while avoiding dynamic as well as static obstacles.
- Moving Holonomically across a feature such as a long shelf, and using Robust Computer Vision Algorithm via the camera to detect required coloured bins from that feature.
- Pick up the detected object and place it in a desired location using a Manipulator (future scope of the project).
- Scalability of the Proposed solution, in case the number of features or the layout of the warehouse is changed in the near future.

2.4 The TC200 Autonomous Mobile Robot

2.4.1 Description

The Autonomous Mobile Robot utilised in this project is the TC200 by TECDRON Robotic Systems, based in France. This robot is a versatile and entirely customizable platform specifically tailored for educational and training purposes. Particular emphasis was placed on the TC200 due to its implementation of mecanum wheels, also known as Swedish wheels or omnidirectional wheels. This feature endows the robot with holonomic motion capabilities, making it exceptionally well-suited for tasks within confined spaces with limited manoeuvring room. Furthermore, the TC200 offers the flexibility to operate with a differential drive configuration if the need arises.

Notably, the TC200 is equipped with mounting provisions and compatibility for articulating robots on its top platform, including models such as the Kuka LBR iiwa, Universal Robot UR10, and Techman Robot TM12/TM14. Although the integration of a mobile manipulator is currently beyond the scope of this project, the capability of this has been proactively accounted for, in the future endeavours. This platform can accommodate a substantial payload of up to 120kg, further enhancing its versatility.

The TC200's locomotion is powered by four brushless electric motors, each equipped with individual electromagnetic brakes. This configuration enables the robot to achieve a maximum speed of 7.24 kph while delivering a formidable torque of 280 Nm. Notably, each motor is outfitted with a high-resolution 2048 ppm encoder, facilitating precise localization and mapping due to its closed-loop operation.

For programming and interfacing, the Robot Operating System (ROS) was used, which provides a robust and highly adaptable software framework. ROS empowers us to enable various autonomous navigation and computer vision functions tailored for industrial applications. However, it's crucial to note that the TC200's software and hardware functions are entirely dependent on ROS. Consequently, a meticulous bit-for-bit external backup to mitigate the risk associated with the potential corruption or unitability of the ROS environment has been implemented.

Autonomous navigation, a critical aspect of the project, necessitates accurate environmental perception. To achieve this, employment of a pair of Hokuyo LIDARS was done, enabling comprehensive scanning and map-building capabilities. Additionally, the TC200 boasts a remarkable operational endurance, capable of running continuously for six hours, thanks to its 51.1V 74.1Ah battery. The TC200 stands as a robust and rugged robotic platform ideally suited for warehouse management applications, aligning seamlessly with the objectives of this specific project [10].

We additionally connected a camera to this AMR for the implementation of Computer Vision Algorithms for the detection of different coloured bins on the shelves of the warehouse and for extending the future scope of the project, where this camera can be used for object detection and other applications. The task of transportation of the detected objects throughout the warehouse, to different locations, is a direct extension to the proposed solution and may come under the future scope of this project as it was not touched upon in this particular project, due to time and other constraints.

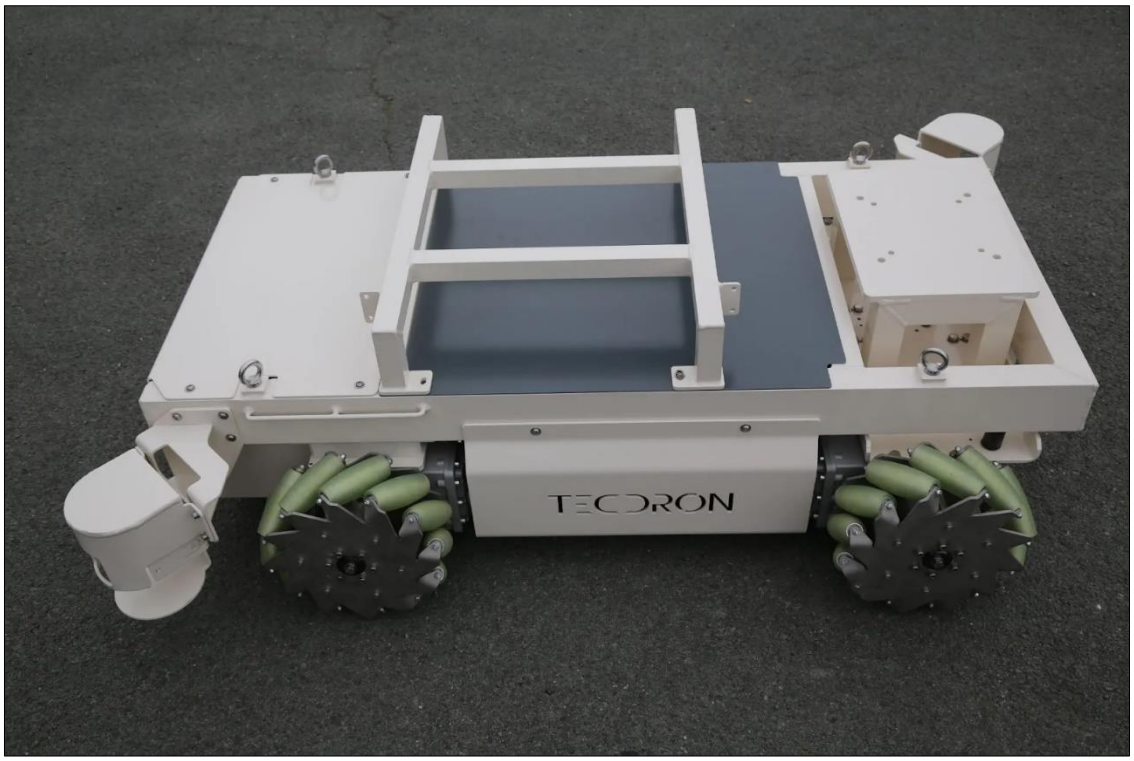

Fig. 2: TC200 Modular Robot

2.4.2 Locomotion

The TC200 robot uses mecanum wheels, which are a specialised type of omnidirectional wheel commonly used in robotics and industrial applications. They feature a unique design with multiple rollers mounted at an angle to the wheel's rotation axis. This configuration allows mecanum wheels to achieve omnidirectional movement, meaning they can move a vehicle or robot in any direction without the need for complex steering mechanisms.

In industrial applications like warehouse management and robotics, mecanum wheels offer several advantages. Their omnidirectional mobility allows for precise and agile movement in tight spaces, making them ideal for navigating crowded warehouse aisles or confined manufacturing environments. They enhance the efficiency of material handling and goods transportation by enabling robots to move sideways, diagonally, and spin on the spot.

Although some draw do consist, such as reduced load capacity, limited traction on certain surfaces, and increased complexity in control algorithms. Moreover, they tend to be more expensive and less energy-efficient than traditional wheels, and their rollers can wear over time, potentially requiring maintenance. However, mecanum wheel-equipped robots excel in tasks such as autonomous inventory management, goods delivery, and assembly line operations, where manoeuvrability and adaptability to changing layouts are critical. Their versatility and ease of control make them valuable tools for improving productivity and automation in industrial settings. Hence, it is perfect for this use case.

2.4.3 Path Planning

Autonomous navigation in holonomic robots like the TC200, which possess the capability to move in any direction without the need for additional rotation, relies on the use of planning algorithms. Planners are software systems that generate a sequence of actions to guide a robot from its current state to a desired goal state.

Planners are commonly categorised into two types: global and local planners. Global planners are responsible for charting an optimal path from the robot's current position to the goal based on pre-existing knowledge of the environment and static obstacles. They employ traditional graph search algorithms and necessitate a precise and accurate map for representing environmental features. However, they do not take into account the robot’s dynamics or sensor uncertainties. Consequently, while they can provide an initial optimal path based on known environmental data, they may not be adaptable to sudden changes or unforeseen obstacles [11].

On the other hand, local planners come into play by dynamically adjusting the initial plan generated by the global planner in real time. They manage the robot's driving dynamics and resolve minor conflicts, such as avoiding dynamic obstacles. Local planners generate both a trajectory and a velocity profile, which are then transmitted to the robot's actuators. These local planners play a vital role in ensuring safe navigation, particularly in dynamic environments like warehouses or factory floors [12].

In the context of ROS, several local planners tailored for holonomic robots were explored. Notably, the "base_local_planner" package within ROS offers parameters specifically designed for holonomic robots, enabling them to leverage their omnidirectional movement capability. Different local planners exhibit varying trade-offs, including computational demands and processing times [13].

The base_local_planner operates by continuously assessing the robot's immediate surroundings, considering factors such as the current robot pose, the global path, and the sensor-derived map of obstacles. Its primary objective is to compute a velocity command that directs the robot safely toward its goal while avoiding collisions with obstacles. Moreover, its ability to generate trajectories that exploit the full range of motion available to holonomic platforms is what makes it especially useful for this project [14].

At the heart of the base_local_planner lies the Dynamic Window Approach (DWA), a dynamic trajectory optimization algorithm. DWA operates by sampling velocities within the robot's control space, encompassing linear and angular velocities. Each sampled velocity is evaluated through a forward simulation, predicting the robot's trajectory over a short duration. The critical decision-making step is the scoring mechanism, where trajectories are assigned scores based on proximity to obstacles, alignment with the global path, distance to the goal, and speed. Trajectories leading to potential collisions are promptly discarded, leaving behind the trajectory with the highest score as the chosen path. This trajectory's corresponding velocity commands are then transmitted to the TC200's mobile base, instructing it on how to navigate effectively [15].

The choice of base_local_planner, specifically leveraging the DWA algorithm, is motivated by its suitability for holonomic robots like the TC200. It harnesses the robot's omnidirectional movement capabilities to swiftly and safely navigate complex environments. Furthermore, its integration within ROS ensures seamless compatibility with other system components, facilitating a robust and coherent autonomous navigation framework.

In terms of global planning, NavFn planner was employed, which offered a swift, interpolated navigation function that can generate plans for a mobile base. This planner operates on a costmap to determine the most cost-effective route from a starting point to an endpoint within a grid.

The navigation function is computed using Dijkstra's algorithm, with potential future support for an A* heuristic. NavFn also provides a ROS wrapper for the NavFn planner, adhering to the nav_core::BaseGlobalPlanner interface as specified in nav_core.

The choice of NavFn as the global planner is driven by its efficiency and effectiveness in path planning, particularly for holonomic robots. This attribute makes it exceptionally useful in intricate environments where the robot may need to navigate around obstacles or through narrow passages, such as warehouses and factory floors.

NavFn operates within a ROS namespace specified during initialization and adheres to the nav_core::BaseGlobalPlanner interface found in the nav_core package. It publishes the most recently computed plan by NavFn each time the planner calculates a new path, primarily for visualisation purposes [16].

Chapter 3: Methodology

3.1 Use Case Overview

A typical warehouse interior is given in Fig. 3 where there are some shelves containing materials and some random boxes placed. In the context of typical inventory management/inventory tracking, in order to get a desired object from the shelf, a warehouse worker (human) will have to go to the desired shelf and look for the object. For a larger warehouse, this implies that the time complexity increases as there will be more workers carrying out similar tasks. This results in an unorganised way of work which is inefficient resulting in more time and resources.

Fig. 3: Use Case Environment for the Concept

In Fig. 3, the schematic view of a shelf is given in the circular image at top. It can be seen that different types of objects are placed on the shelf. The goal is to use the TC200 robot to find objects which are of interest to the warehouse worker. The robot is placed at a parked position P. When the warehouse worker needs to track/find an object of interest from the shelves, the robot helps to ease the task by going to the shelves marked with numbers (1...5) autonomously and then looking for the desired object. This saves time for the worker and also a co-dependency between the robot and human is established. The operation of the robot can be visualised in Fig. 4.

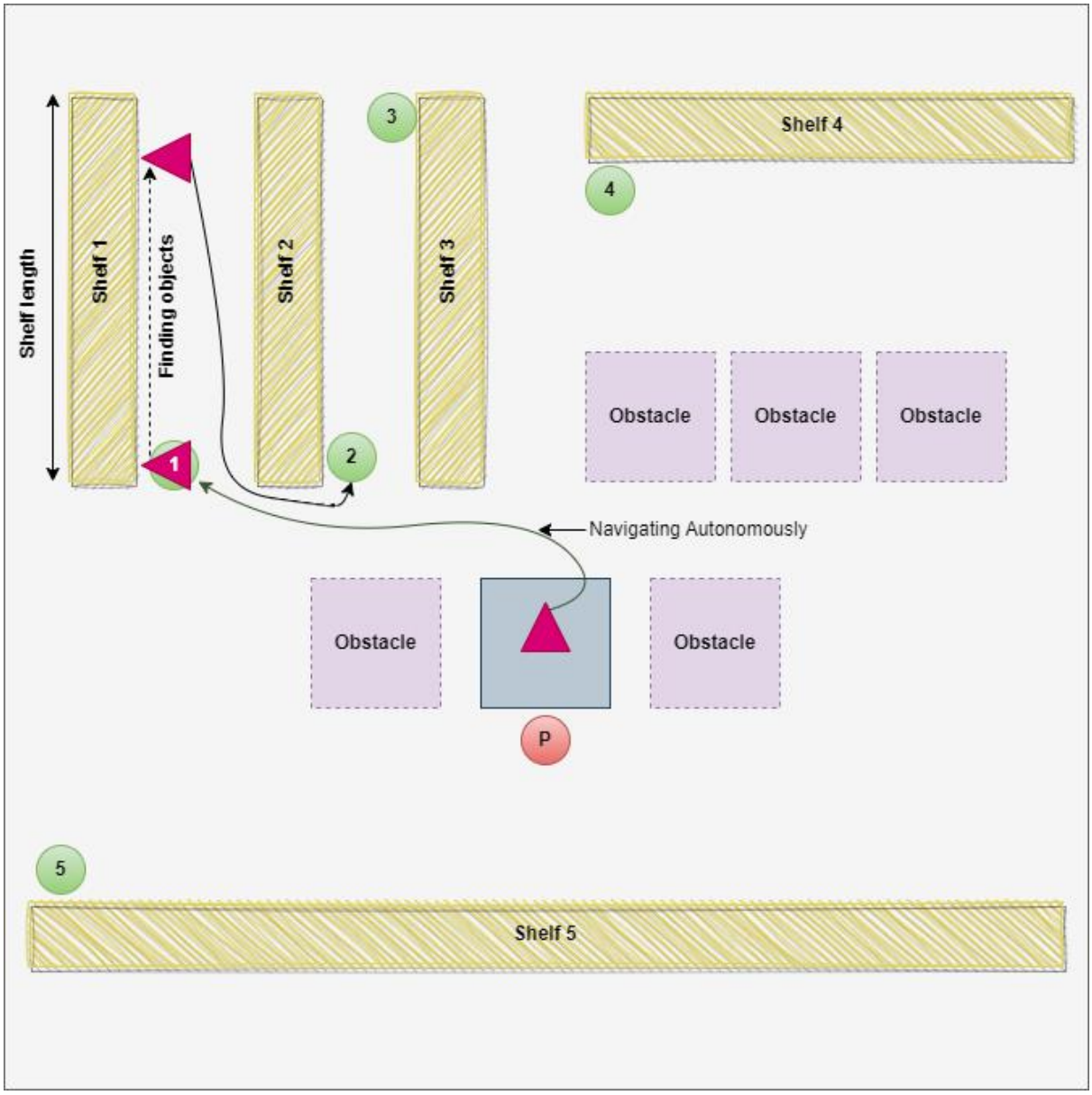

Fig. 4: Operation of the Robot

The robot starts from the parked position and then autonomously navigates to the shelf point 1. After it reaches there, it starts moving parallel to the shelf and simultaneously searches for the desired objects. Whenever an object is found, the robot stops momentarily before continuing again along the shelf. This process continues until the robot has reached the shelf end. After that, the robot moves on to the next shelf autonomously and performs the functionalities again. The process of navigating to a shelf autonomously and then searching for the objects along that particular shelf until reaching the end of that shelf can be regarded as a task. After the completion of all tasks, the robot returns to the parked position using autonomous navigation.

Based on the specific use case application, the two main features are:

Autonomous Navigation:

This feature allows the robot to autonomously navigate between two given map coordinates while avoiding obstacles. This is done using the autonomous navigation function which takes the destination coordinates as inputs and then uses a global planner to plan a route to the destination coordinate. After that, the module employs a local planner to execute the global path while avoiding both dynamic and static obstacles. The Autonomous Navigation is also kept as a separate functionality to allow the robot to navigate to any random location on the map. This function is helpful to determine the shelf locations which will be further discussed in Chapter 4: Implementation.

Object Tracking:

The primary objective is to find objects of interest from a shelf. The shelf could be located at a certain location in the warehouse. The robot is at the parked position. Therefore, at first, the robot employs the Autonomous Navigation to ensure that the robot autonomously navigates to the starting point of the shelf. The shelf starting points are provided by the robot operator which acts as the input of this feature. After the robot has reached the shelf, the Object Tracker feature starts and moves the robot holonomically such that the robot is perpendicular to the shelf and is simultaneously moving parallel. The Object Tracker feature also employs some computer vision algorithms to find the objects on the shelf.

The motion of the robot is put to a halt if either of the three scenarios mentioned below occurs:

1. The Robot finds an object on the shelf and hence the holonomic motion is stopped momentarily to further assess information.
2. The robot detects an obstacle in the direction of holonomic motion and stops immediately. The motion will continue only when the obstacle has been cleared.
3. The robot reaches the end of the shelf and stops.

In the next segment, which is the system architecture, a more detailed explanation of each feature along with its sub-modules and their functionalities has been presented.

3.2 System Architecture

The features of the proposed concept design for the use-case have been further divided into sub-modules and are presented in Fig. 5, giving a topological idea of the important modules that are used.

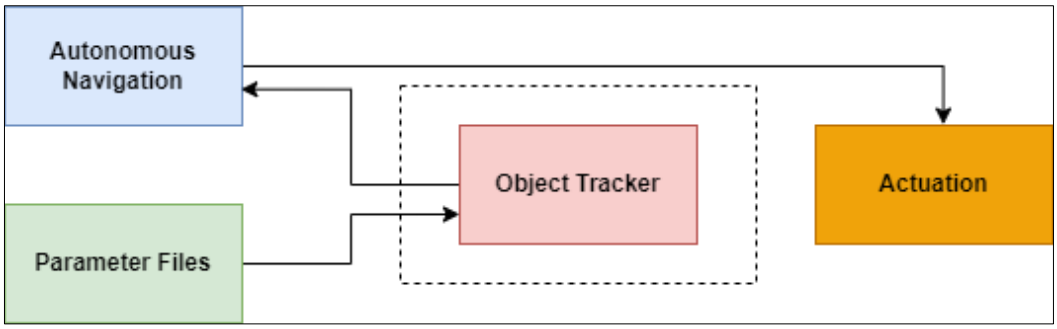

Fig. 5: System Architecture for Concept Design

3.2.1 Parameter files

User-defined files that contain the "directions" for the robot to read and execute. The parameter files contain the information with respect to each shelf in the warehouse. The robot is provided with these files as input which is then read by the "Task Scheduler". The "Task Scheduler" executes the tasks in a sequential order. More details about the structure content of the parameter files have been provided in Chapter 4: Implementation.

3.2.2 Autonomous Navigation

To accomplish this goal, the robust features offered by ROS (Robot Operating System) were leveraged. For autonomous movement from one point to another, the robot employs a navigation stack, with "MoveBase" [17] serving as its central component as shown in Fig. 5. "MoveBase" integrates multiple algorithms and modules to present a unified navigation interface.

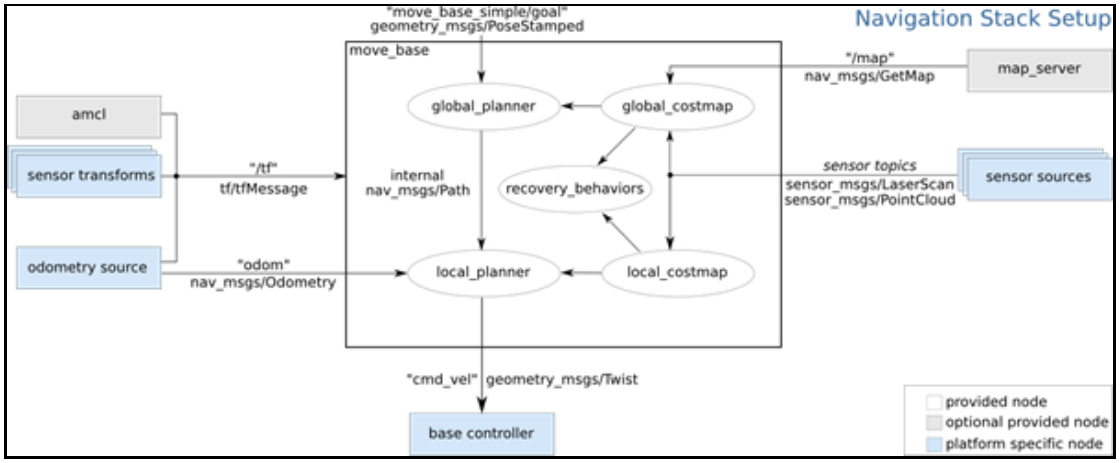

Fig. 6: move_base node overview

Fig. 5 provides an overview of the move_base node and its interplay with various components. Within MoveBase, there is a local planner that interfaces with diverse robot localization tools such as AMCL and different array of sensors in IMU (Inertial Measurement Unit for precise robot positioning and odometry data. Additionally, the local planner plays a crucial role in guiding the robot around obstacles encountered along the way to its intended destination. Furthermore, as previously explained the "global costmap" component receives map data from

the “map_server”, aiding the robot in charting its course to the designated goal location. The global planner operates in accordance with the information contained in the “global costmap”, assisting the robot in its journey to a specific destination.

Global and local planners play a crucial role in performing autonomous navigation by obstacle avoidance. A global planner is needed to find a feasible route that considers the overall structure of the environment. For that, global planners use different path planning algorithms like the grass-fire algorithm, Dijkstra Algorithm, A* algorithm, and many more. The local planner operates at a lower level and is responsible for making real-time adjustments to the robot's path to ensure safe and collision-free movement. It considers the robot's immediate surroundings and dynamically adapts the path as necessary. It is responsible for generating specific control commands like velocity and steering by avoiding dynamic obstacles or unexpected changes that may not be considered in the global planner. This helps robots to navigate around sudden threats.

As a local planner, the robot uses a DWA (Dynamic Window Approach) planner which allows the better holonomic motion of the robot and as a global planner, Navfn is used which utilizes the Dijkstra path planning algorithm as this planner strictly adheres to the global plan and leading to accurate positioning and navigation.

Hence, the robot gets the information (x,y, and θ) from the user according to the robot coordinate system and moves at the prescribed location in the environment autonomously by avoiding obstacles.

3.2.3 Object Tracker

As discussed in the use case overview, the robot reaches the shelf start location by Autonomous Navigation based on the provided data of (x,y, and θ) by the user. The process overview of the object finder function is depicted and explained in Fig. 6:

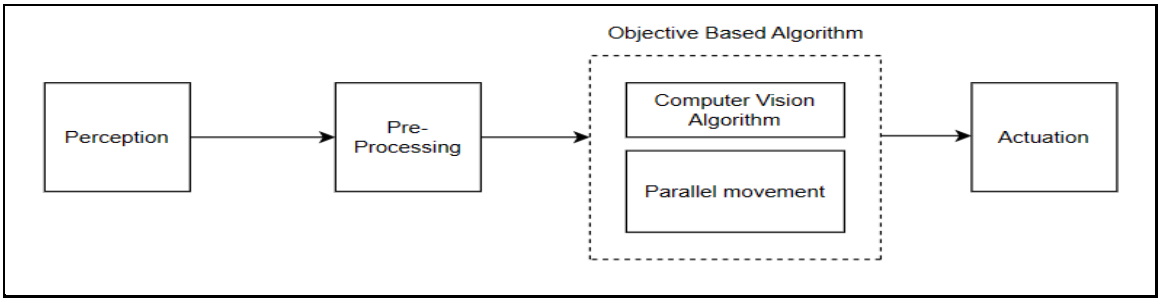

Fig. 7: Fundamental logic of the Object Tracker

3.2.4 Perception

A pair of perception sensors integrated into the robotic system, were utilised for perception purposes. The initial sensor in the setup is a LIDAR sensor which provides one-dimensional data allowing us to ascertain the proximity of objects within the robot surroundings and the second sensor is a camera that captures the images of the real environment.

3.2.5 Pre-Processing

These perception sensors are prone to noise. So, to get the effective data, pre-processing is an essential part.

- **Lidar:** Incorporating the lidar data in the fitting algorithm involves a transformation process where it is converted into the point cloud data which gives the 2-Dimensional representation of the lidar scan data and converts it into the robot coordinate frame to ensure x and y values are consistent to feed in the algorithm. More information is provided in section 4.3.
- **Camera:** For computer vision, the images are acquired from the camera and then pre-processed and then used to feed into the algorithm. More information is provided in section 3.3.

3.2.6 Objective-Based Algorithm

In this part of the process, based on the objective of both sensors the algorithm is provided.

- **Fitting Algorithm:** The pre-processed data of lidar sensor is introduced into the fitting algorithm known as RANSAC which is used for robust estimation of model parameters from noisy data allows the removal of noise from the data and fitting the data accurately by providing the fitting line, its slope and intercept. The visual representation of the fitting line generated by the algorithm is illustrated below:

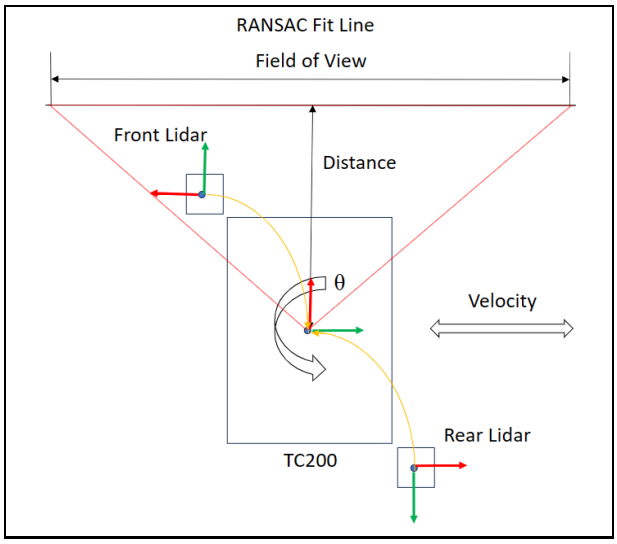

Fig. 8: General idea of RANSAC algorithm

In the above Fig. 7, it is described that when TC200 reaches in front of the shelf, the RANSAC algorithm comes into play, generating the fitting line based on the estimation model leveraging the data collected by the lidar sensors. This data is then used to facilitate the alignment of the robot with the wall by using P-Controller. The robot achieves parallelism with the wall exclusively when the vector ‘x’ within the robot's coordinate system is orthogonal to the vector ‘y’ within the map coordinate system.

- **Computer Vision Algorithm:** Camera data after preprocessing is used in the detection algorithm to effectively detect the objects. Based on the results from the algorithm, the robot achieves the capabilities of detecting the objects and initiating a momentary stop. As discussed in part 4.4.

3.2.7 Actuation

Based on the fitting data, the robot performs its parallel motion along the shelf with the help of two proportional controllers where one controls the distance of the robot to the shelf (x-direction) and second one the orientation of the robot (θ-yaw angle) with respect to the shelf. Based on the computer vision algorithm, if the objects are detected and the robot is stopped, it displays the information of the number of objects detected and detection accuracy to the user.

This complete workflow is separated into three modules which can be referred to as the system architecture of the entire function presented and explained in Figure 8:

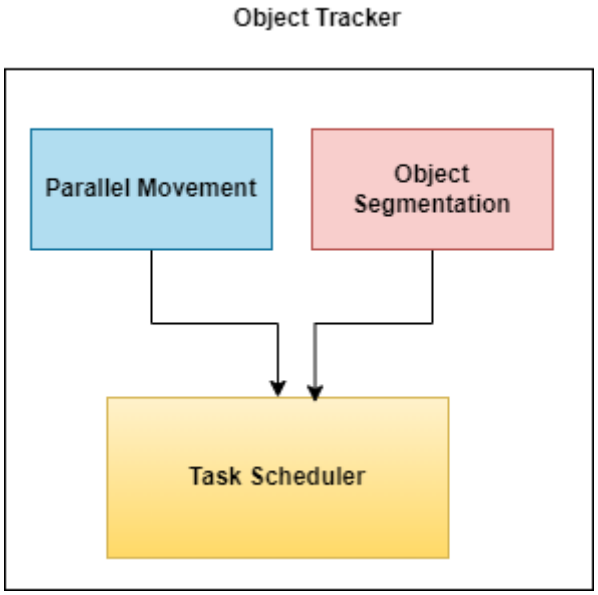

Fig. 9: Object Tracker Architecture

- **Parallel Movement**

  This module is responsible for moving the robot parallel while facing the shelf as well as the implementation of different stop conditions based on the situation. This module generally uses the RANSAC algorithm and a P-Controller to initiate the parallel motion.

- **Object Segmentation**

  This module helps robots detect and identify the objects stored on the shelf while the shelf follower module is in active mode. The complete module has been explained well in section 3.3.

- **Task Scheduler**

  This module enables robots to execute the two previously described modules for a specified number of tasks provided within this module.

3.3 Computer Vision and Deep Learning Model

This chapter delves into the foundational pipeline of a deep learning model and outlines the various critical steps involved in the process. It provides a comprehensive discussion on how to execute these steps effectively.

3.3.1 Pipe Line of a Deep Learning Model

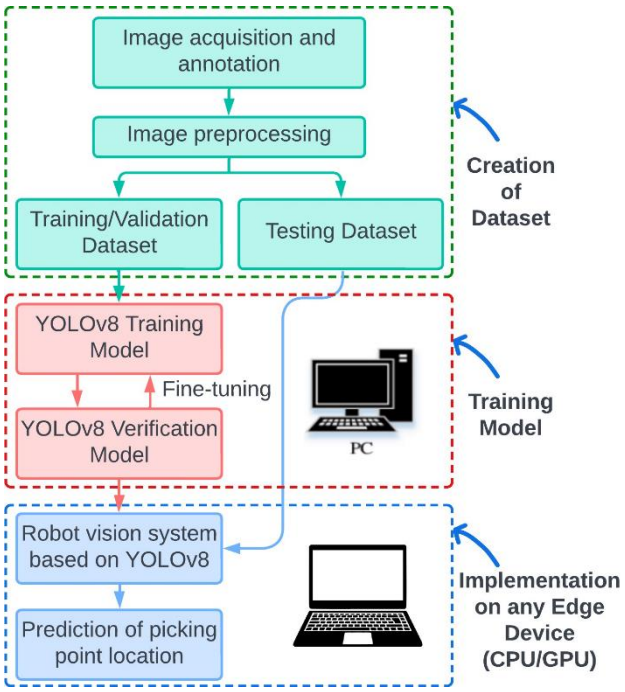

Fig. 10: Basic pipe line of a deep learning model in Computer Vision

3.3.2 Creating a Custom Dataset

A custom dataset in computer vision is a dataset that is specifically created for a particular application. This means that the images or videos in the dataset are relevant to the specific task that the computer vision model is being trained to perform. If custom data sets are not available online, then in most cases it has to be generated from scratch.

- **Image Acquisition**

  Dataset generation involves capturing or sourcing relevant images that represent the specific task or scenario the dataset is intended for. This process may involve using

cameras, downloading images from the internet, or utilising other data collection methods.

Points considered while taking images:

1. Variety of images: Different lighting, Angle of view, the background and the pose of the object
2. Quality of image: Clear and sharp images were taken to highlight the key features of the image. Images with some level of noise or low-quality images were also taken to imitate the real-world scenario of real-time image acquisition. In Machine Learning training "Garbage In is Garbage Out".
3. Adequate quantity: The greater number of images the better the model will perform.

- **Data Annotation**

Data annotation is the process of adding human-readable labels to data. This can be done for a variety of purposes, such as training machine learning models, creating search indexes, or making data more accessible to people with disabilities

a. Manual Annotation of Image: Use of manual annotation tools for creating boundary boxes or Polygons or masks on the object to be annotated.
Notable tools: LabelMe, LabelImg, VGG Image Annotator

b. Automatic Annotation:

i. **Supervised Annotation**: It is the process of manually labelling images, and then using those labels to train a machine learning model to automatically annotate new images. This is the most common approach to image annotation, and it is the most accurate way to get high-quality annotations. However, it is also the most time-consuming and expensive.
Notable tools: CVAT, Hasty.ai, YAT, AutoML Vision Edge

ii. **Fully Automatic Annotation**: It is the process of automatically annotating images without any human intervention. This is a newer approach to image annotation, and it is becoming more and more popular as the accuracy of machine learning models improves. However, fully automatic annotation is not yet as accurate as supervised annotation, and it can be difficult to get high-quality annotations for complex images. Deep learning models can be used for fully automatic annotation of the image such as Segment Anything Model.

- **Data Augmentation**

Image data augmentation is used in machine learning to improve the performance of models trained on image data. By artificially increasing the size and diversity of the dataset, data augmentation can help models learn to better generalise to unseen data.

Reasons of doing Data Augmentation:

1. Reduces model overfitting
2. Makes the model robust to variation
3. Reduces manual data annotation
4. A new dataset can be created from the existing dataset

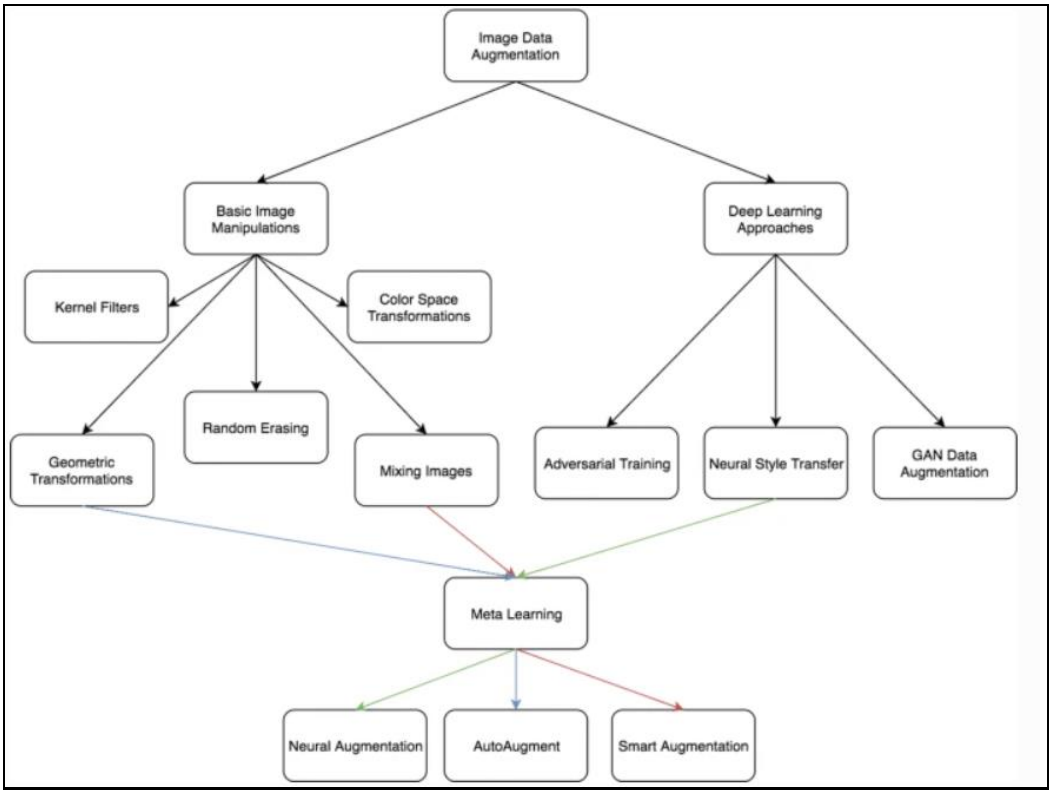

Fig. 11: Various methods of Data-augmentation

3.3.3 Training a Deep Learning Model

There are various methods of training a Deep Learning Model: -

1. **Training from Scratch**

   This is the most common method and involves training the model from scratch on a dataset of images with ground-truth annotations. This can be a time-consuming process, but it can also lead to the best results.

2. **Fine-Tuning**

   This method involves starting with a pre-trained model and then fine-tuning the model on a dataset of images with ground-truth annotations. This can be a faster way to train a model, but it may not lead to the best results.

3. **Transfer Learning**

   This method involves starting with a pre-trained model from another task, such as image classification, and then fine-tuning the model on a dataset of images with ground-truth annotations for the new task. This can be a very efficient way to train a model, but it may not be as accurate as training from scratch.

4. **Ensemble Learning**

   This method involves training multiple models on different datasets and then combining the predictions of the models to get a better overall prediction. This can be a very effective way to improve the accuracy of a model, but it can also be a more complex and time-consuming method.

Why using transferred learning for training a Yolov8 model on a custom dataset is better:

- It saves time and resources. Training a model from scratch can be time-consuming and computationally expensive, especially if you have a small dataset. Transfer learning can help you to save time and resources by reusing the knowledge that has already been learned by a pre-trained model.
- It improves accuracy. A pre-trained model has already learned to extract features from images and to classify objects. This knowledge can be transferred to your custom dataset, which can help to improve the accuracy of the model.
- It makes the model more generalizable. A pre-trained model has been trained on a large dataset of images, which helps it to generalise to new images that it has not seen before. This is important for object detection, as you will often need to detect objects in images that are different from the images that the model was trained on.

3.3.4 Deployment of Deep Learning Model on a Mobile Robot

The Trained Deep Learning model is validated and deployed on the Autonomous mobile robot. The entire process is orchestrated using the Robot Operating System (ROS). Real-time inference is achieved through efficient hardware resources integrated into the mobile robot. The trained deep learning model, specifically a YOLOv8 architecture, is fine-tuned on a custom dataset, tailored to the unique demands of the warehouse scenario.

This approach not only leverages the pre-trained knowledge but also ensures the model's generalizability to handle diverse objects and environments. The ROS-based deployment allows seamless communication between various components of the robotic system, ensuring effective execution of object detection and tracking tasks in real-world warehouse settings.

Chapter 4: Implementation

4.1 Use Case

In the context of implementation, the modules mentioned in Chapter 3 were developed and used to detect and identify bins from the shelf. For the proof of concept, only the bins of red and yellow colour were detected. When the robot is performing a task, when it detects and identifies an object of interest on the shelf, it displays some metadata that is relevant to that respective object such as the confidence score and the number of occurrences of that object. Fig. 11 shows the system diagram for implementation purposes.

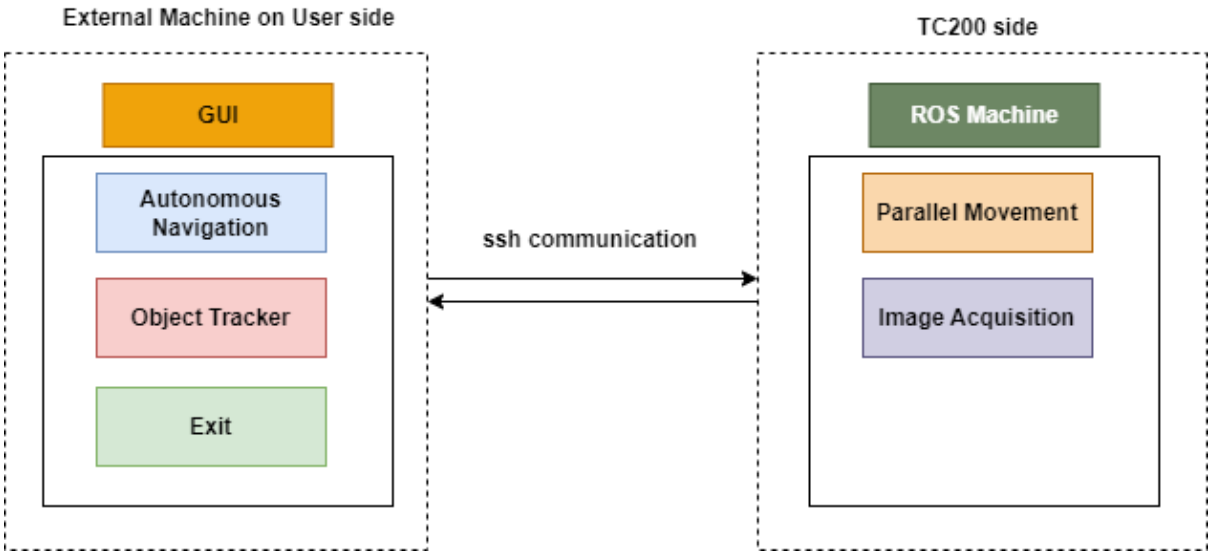

Fig. 12: System Architecture for Implementation

The overall system has been split into two perspectives:

- External Machine that is running on the User side which runs the modules: Autonomous Navigation, bin tracker and task scheduler
- TC200 Robot side which runs the modules: state machine and image acquisition

The functionality of the modules described in Chapter 3 remains the same. The reason for this specific arrangement in the system diagram is to assert the fact that the system can be scalable and run on different user machines. Hence this gives the user a degree of freedom in choosing which programs or modules to run on the robot depending on the requirement. As two separate ROS machines were running, the communication between the machines took place through SSH (Secure Shell) protocol [18]. The GUI which is run on the user side wraps the modules discussed in Chapter 3.

Fig. 13: Graphical User Interface for Use Case

The basic functionality of GUI according to Figure 13 is defined below:

1. **Autonomous Navigation**

   When invoked, asks the user to input x, y and theta of the target location on the map. The robot then navigates autonomously to the target location while avoiding obstacles. This function is useful if the robot has to travel to a non-shelf coordinate in order to perform other tasks.

2. **Object Tracker**

   When invoked, it performs the scheduled tasks mentioned in Fig. 9.

3. **Exit**

   When invoked, quit the application.

The next segment gives a detailed explanation of the implementation of the modules wrapped by the GUI.

4.2 Autonomous Navigation

The overview of the Autonomous Navigation can be obtained by referring to section 3.2.2. To implement the Autonomous Navigation, it is necessary to follow a series of steps, which can be elucidated as follows:

4.2.1 Map building or Mapping

Before implementing the Autonomous Navigation, the prerequisite of mapping must be fulfilled. For that reason, the built-in ROS package “Gmapping” is used which uses the data from LIDAR scan. The robot is manually driven with a joystick while the gmapping function

is active. The environment is explored and a 2D map is formed until the final results are satisfactory. This map is later used by the robot as a reference to localise itself using the AMCL.

4.2.2 Acquiring goal locations

Autonomous navigation consists of several parts which allow robots to reach specified locations. One of the major important parts is the AMCL package. This package is responsible for estimating the robot's position and its orientation within the map. It continuously updates the robot pose estimates as it moves.

To acquire the AMCL pose, it is necessary to do mapping as it is an essential part which provides robots with fundamental knowledge of the environment. With mapping only AMCL can get the reference to compare the sensor data.

To get the robot's current location into the environment, AMCL was used, which publishes the robot's position (x,y and z) and orientation (x,y,z and w) which is in Quaternion angles with respect to the map frame in the topic "amcl_pose". The position of the robot (x and y) was calculated from the amcl position data but to get the orientation of the robot in an understandable manner, the conversion of the data from Quaternion to Euler Angle (θ) was required. These coordinates of the robot (x,y and θ) represent the robot's location in the actual environment. The robot travelled to different locations on the map, in order to get its coordinates on a specific location.

4.2.3 Travel to the goal location

After acquiring the coordinates of the goal location, the robot uses the ROS navigation stack to reach the destination which is well explained in section 3.2.2.

4.3 Object Tracker

As discussed in the methodology section, the function consists of two modules: Task Scheduler, and object tracker. The Module name Task Scheduler consists of different sub-modules which are Autonomous Navigation and State Machine.

4.3.1 Task Scheduler

This module has been designed to enable the robot to operate in various shelf start locations and execute the Autonomous Navigation and Parallel modules by invoking them by setting the parameter "enabled 1". The parameter files are provided to give the capabilities of performing these modules on different shelf locations.

What are Parameter Files?

The parameter files are provided by the user which provides the task description to the robot. The parameter files use the JSON (JAVASCRIPT OBJECT NOTATION) format and 5 specific information with regards to the shelf is stored.

The 5 key pieces of information are: -

- x-coordinate of shelf start location
- y-coordinate of shelf start location
- theta angle of shelf start location
- shelf length: This parameter also acts as a distance indication for the robot to travel along the shelf. The user provides this length of the shelf in meters.
- Direction: This parameter indicates whether the robot should travel towards right or left.

The parameter files are stored on the user side and by providing these 5 crucial pieces of information to the robot, the task can then begin.

The sub-modules of the Object Tracker module have been explained below:

4.3.2 Parallel Movement

This module manages the robot's lateral movement along the shelves. The section delves into the specifics of its implementation and the techniques employed. A detailed explanation of the implementation of the RANSAC algorithm, P-Controllers, stop conditions tailored to various robot states, and situational factors is provided below.

RANSAC Fitting Algorithm

The robot is equipped with two Lidar sensors, strategically positioned at its front and rear ends. In real-world scenarios, Lidar data can be susceptible to noise, which may jeopardise task execution. To mitigate the risk of failure and render the data suitable for analysis, a crucial pre-processing step is undertaken. During this pre-processing phase, the data is typically transformed to align with the map frame's coordinate system, which inherently differs from the robot's coordinate system. This alignment enhances data utility. Subsequently, the Lidar data is converted into the robot's coordinate frame for optimal usability.

The Lidar sensor captures data across a range of 0 to 719 data points. However, for the specific task of projecting the fitting line onto the shelf, only a subset of these points is required. To achieve this, the Lidar data is meticulously filtered and selected based on its relevance, and then it undergoes a transformation into a 2D point cloud format with x and y coordinates. This refined data format enhances its efficiency and suitability for subsequent algorithmic analysis, where it is used as input.

The implementation of the RANSAC algorithm is facilitated through the utilisation of the "Scikit Learn" machine learning library. This implementation involves providing the algorithm with 2D point data in the form of x and y coordinates as input. The RANSAC algorithm, in conjunction with the "Scikit Learn" library, is designed to effectively derive the fitting line by eliminating extraneous data points that are not relevant to the task.What sets this implementation apart is the automation of certain critical aspects. The RANSAC method provided by the "Scikit Learn" library is capable of autonomously determining the appropriate number of iterations and identifying the inliers for the algorithm. This automated process significantly reduces the complexity of parameter selection.

Ultimately, the RANSAC algorithm, when executed with the "Scikit Learn" library, yields the fitting line, complete with its slope and intercept. These output values are subsequently employed to orchestrate the parallel motion of the robot along the shelf, aligning it with

precision. The explanation and the visualisation of the RANSAC algorithm can be understood by the Fig. 8.

Proportional Controller

The Lidar data is always changing which results in sudden fluctuations in the results of the RANSAC algorithm. This can cause variations in the slope and intercept data which leads the robot to have jerks in its motion while performing the tasks. To eliminate the jerks and control the distance between the robot and wall (x) and the orientation of the robot (θ) two P-Controllers have been implemented. One controller aims to correct the error arising from the disparity between the target distance from the wall and the intercept. Similarly, the second one aims towards the control of robot orientation (θ) where the error definition is the difference between the target orientation of the robot which is predefined as 0.0 and the slope. The wheel movement is initiated based on this via the "cmd_vel" topic where the velocity limits in the x,y and z directions of the robot were provided.

The mathematical representation of the P controller is elaborated upon in detail in section 5.1.2. Within this section, you will find a comprehensive explanation of the controller's mathematical formulation and its role in the robot's navigation and control.

Additionally, various stop conditions have been meticulously implemented to cater to diverse scenarios and situations. These conditions enable the robot to make informed decisions about when and how to halt its motion based on the specific circumstances it encounters during its navigation tasks. This has been explained in the following section.

Stop Conditions

During the task execution, there can be some situations where the robot needs to be stopped or halt its movement for a while. These situations have been explained in the section 3.1 with Object Tracking. To implement these stop conditions, the Lidar data and the Camera data are used. To implement the stop condition 2, Lidar data is used where if any object comes near the specified range of the robot, 0.0 velocity is published in the "cmd_vel" topic. In the stop condition 3, as in the parameter file the shelf distance is already provided. If the shelf distance is already reached, the robot stops its movement. Stop condition 1 is implemented based on the computer vision algorithm. Robot stops momentarily if the bin is detected in the camera and continues its motion after 10 seconds.

Object Segmentation

This module consists of a computer vision algorithm which provides the mind to the machine to detect the bins placed on the shelf. Image acquisition runs on the robot's computer, takes the image of the environment and publishes in the topic which the computer vision algorithm then subscribes for data processing. The detailed implementation of this module has been explained in further sections of 4.4.

4.4 Computer Vision and Robot Perception

In this section the entire pipeline following which the instance segmentation model was trained and deployed later on a separate computer, is discussed. When the robot moves parallel to the

wall during Object Tracker, the algorithm detects the bin and segments it by applying a mask on it, and a tracker function keeps a count of the number of bins.

4.4.1 Pipeline of the Instance Segmentation Model

This section gives the pipeline of how the Yolov8 segmentation model is trained and for using it as an object identification module on the TC200 robot's Object Tracker function. The complete pipeline used to train the Instance segment model is described in the Fig. 13.

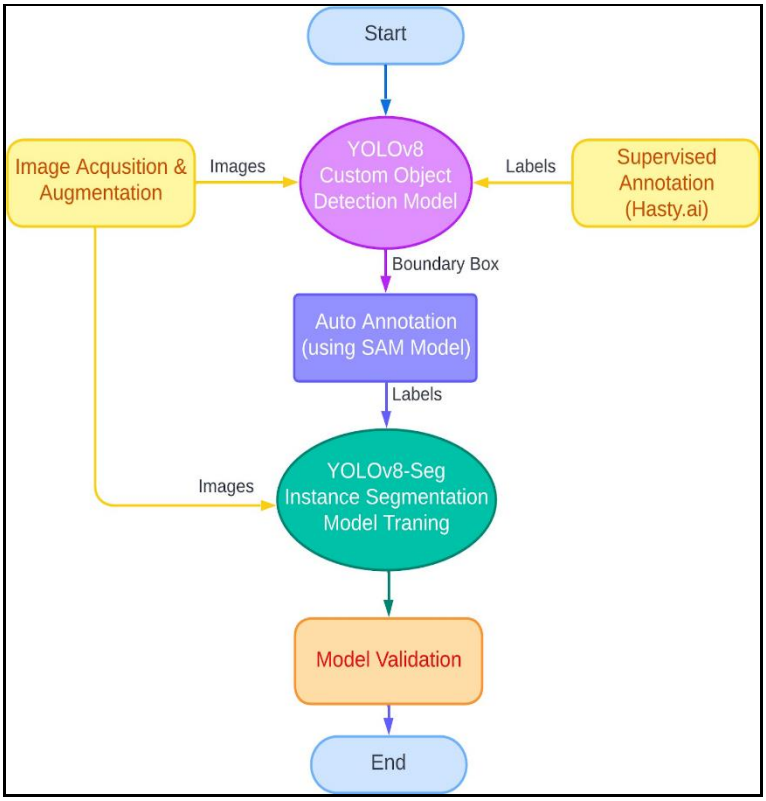

Fig. 14: Complete pipeline of the implemented Yolov8 Model

The Segment Anything Model (SAM) is a large language model (LLM) developed by Meta AI that can segment every object in an image with high accuracy. This makes the model quite large and the computing power required to operate it in real time is quite expensive.

As an alternative, the Yolov8 model is used for instance segmentation since this model gives a considerable amount of accuracy and real-time operating speeds even with CPU.

The input to the SAM model is a boundary box from the Yolov8 Object detection model. The SAM model gives the corresponding instance segmented mask in txt format, which is later used for training the Yolov8 Instance segmentation model. Annotation for instance segmentation tasks is quite tiresome and costly, thus this implementation makes the process cost and time-effective.

4.4.2 Custom Dataset Generation

- **Image Acquisition**

  In this project, various methods of image acquisition were implemented, most images were taken from a phone camera, and some images were taken from a webcam of the robot to imitate the actual perception of the mobile robot. Other images were downloaded to create a varied data set.

Images acquired	300 in total
Object to be detected	Industrial storage bins
Class	2 Class: [Red bin] and [Yellow bin]
Key feature	Colour, Shape, Size and Orientation
Shortcoming	No unique pattern or texture on the surface. Only difference between two bins is colour

- **Data Annotation**

  In this project, two different types of annotation were done according to the task.

  a. **Boundary box Labelling**: For object detection model

     Use the Hasty.ai tool for supervised annotation of Images in .xml format which is later converted to YOLO format

  b. **Segmentation Mask**: For instance, the segmentation model

     The Segment Anything model is employed for swift and precise annotation of masks on images. It utilises the boundary box generated by the Object Detection model as input to define a Region of Interest (ROI) for tasks related to instance segmentation.

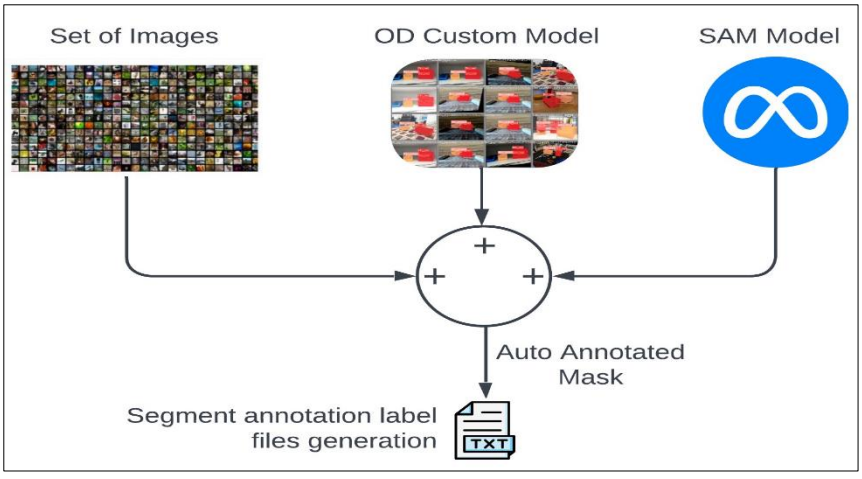

Fig. 15: Pipeline of Segment Anything model for Automatic annotation

A detailed comparison of various Manual and Supervised annotation tools is in the Appendix A1.1 and A1.2

- **Implemented Data Augmentation**

  Data Augmentation:

  In this project taking thousands of images of the Bins [Objects to be detected] was not a viable option. Thus data augmentation was implemented in two different ways.

1. **Data Augmentation to increase the data set**

   It is the process of creating new data points from existing ones and adding them to the dataset. This can be done by applying the same transformations that are used to augment the dataset during training, or by using more creative methods such as generating synthetic data.

Problems	**Reasons**	**Data augmentation method**
Lighting conditions	The lighting conditions in a warehouse can vary greatly, depending on the time of day, the weather, and the location of the warehouse.	Hue, Brightness, contrast (Image enhancement)
Different poses	The same object can be in many different poses in a warehouse. For example, a box can be standing upright, lying on its side, or even upside down.	Rotation, translation, scaling, cropping, mirroring (Geometric transformations)
Fewer images	The number of images available for training a mobile robot can be limited. This is especially true for objects that are not common in warehouses. Also collecting 1000s of images is a tedious process.	Flipping, cropping, noise addition, rotation, translation, scaling
Occlusion	Objects in a warehouse can often be occluded by other objects. This can make it difficult for a mobile robot to see the object and identify it.	Blurring, masking, inpainting
Object size	The size of objects in a warehouse can vary greatly. This can make it difficult for a mobile robot to accurately perceive small objects.	scaling and cropping

Table 2: Use of various Data augmentation along with the reasons

The tools used for data augmentation in this project are **Roboflow** and **Albumenation**.

Data Augmentation during the Runtime

Data augmentation during runtime is the process of generating new data points on the fly, as the model is being trained. This can be done by applying the same transformations that are used to augment the dataset during training.

1. hsv_h: This hyperparameter controls the amount of hue augmentation that is applied. Hue augmentation changes the hue of the images, which helps the model to learn to recognize objects under different lighting conditions.
2. The copy_paste hyperparameter in YOLOv8 controls the probability of applying the copy-paste data augmentation technique. A simple Copy-paste is a strong method of data augmentation for instance segmentation [19].
3. It is a technique that copies and pastes a portion of one image onto another image. This can be used to create new images that contain objects in different poses and locations.

Command to Train the model in Google Colab environment: With data, augmentation parameters passed as arguments:

!yolo task=segment mode=train model=yolov8s-seg.pt data=..//Path to custom yaml file// epochs=35 imgsz=640 copy_paste=0.7 hsv_h=0.2

4.4.3 Training of Yolov8 model on Custom Data

Yolov8 model is trained for two specific tasks in this project:

1. Object detection: This model is trained to give Boundary box of the objects as input to the SAM model
2. Instance Segmentation: This model is trained to do real-time instance segmentation of the objects (industrial storage bins) by the mobile robot.

Steps Involved in training the Yolov8 model:

1. Install the package named “ultralytics”
2. Organise the folder structure for:

    a. Training dataset [70-80% images of the entire dataset with its labels]
    b. Validation set [20-30% images of the entire dataset with its labels]
    c. Test dataset [Images / Videos /real-time testing]
    d. Configuration files: It contains the path to these data set and declaration of the objects to be detected in class in a yaml file

    path: (dataset directory path)
    train: (Complete path to dataset train folder)
    test: (Complete path to dataset test folder)
    valid: (Complete path to dataset valid folder)

    #Classes
    nc: 2# update for the total number of classes to be detected

#classes names
#add new class names here
names: ['Red Bin', 'Yellow Bin']

#various default hyperparameters can be declared and modified
Hyperparameter:

3. Training process

yolo task=seg mode=train model=yolov8s-seg.pt data=custom.yaml epochs=35 imgsz=640

task = segment [could also be detect or classify]
mode = train [could also be predict or val]
model = yolov8s-seg.pt (small) [could also be nano, medium or large model]
epoch = 35 (depends on dataset and model result)
imgsz = 640 (image size: multiple of 32)

How to Increase the Class dataset?

In order to detect more objects increase the class dataset was required, which was quite a straightforward process:

1. **Collect New Data**

   Collect images that feature objects from the new class. If possible, capture images that include both old class and new class objects. If not, employ data augmentation techniques to blend old and new class objects together.

2. **Annotate Data**

   Annotate the new class's objects in the images by drawing bounding boxes (for Object Detection). For Instance, segmentation, and mask annotation by SAM model can be used.

3. **Update Class List**

   Modify the YOLO configuration file to include the new class in the class list, and add the class name to the names file.

4. **Combine Datasets**

   Integrate the new class's annotations and images with your existing dataset while preserving the YOLO annotation format.

5. **Retrain the Model**

   Continue training your YOLO model using the combined dataset, starting from scratch or fine-tuning the existing model with the new data.

6. Evaluate and Tune

After training, assess the model's performance on a validation set, making necessary adjustments to hyperparameters for optimization.

7. Test and Deploy

Assess the model's performance with the new class obe declared test dataset, and if successful, deploy it for desired applications.

4.4.4 Deployment on the Robot

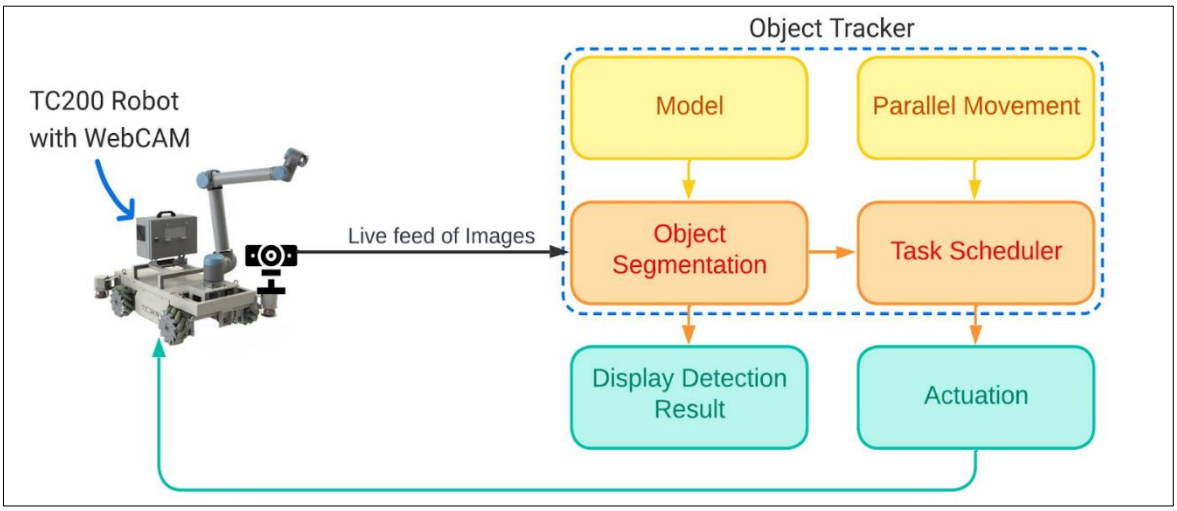

Fig. 16: Communication of Object tracker with the Robot

- **Deployment on the TC200 Robot**

  The Object Tracker algorithm operates on a distinct computer on the user side quipped with the custom Yolov8 Instance segmentation model. The TC200 Robot is linked to this computer through an SSH communication protocol within the same network. Communication between the Robot (TC200) and the external computer follows a publisher-subscriber model, facilitating the transmission of images from the camera mounted on the robot to the user computer.

  The computer, where the Object Segmentation algorithm and Object tracker functions are active, dispatches an actuation signal to the robot to execute a stop command upon detection of the desired object.

- **Object Tracker function**

  For counting multiple objects in the image and to track if the object has crossed a particular ROI in the image a SORT tracker is implemented. SORT is a simple online and real-time tracking algorithm for 2D multiple object tracking in video sequences.

This function keeps and updates the counter for total instances of Object detection, red or yellow bin. As the robot moves parallel to the shelf the objects in the image move backwards (relative to the robot) thus crossing a centre ROI of the frame. This updates the tracker function and gives a stop command to the robot.

Chapter 5: Validation

5.1 Autonomous Navigation Modules

5.1.1 Validation of Laser Scan

The TC200 robot, in this specific use case, is meant to be used in industrial applications like warehouse management. Here, the LIDARs on the TC200 are deployed for various tasks such as obstacle avoidance, path planning, mapping, etc. These applications rely on the precise and reliable data provided by LIDAR sensors. Standard deviation plays a critical role in assessing the quality and reliability of the data obtained from these systems.

LIDAR sensors emit laser beams and measure the time it takes for these beams to bounce off objects and return to the sensor. By analysing the time-of-flight or phase shift of the returning laser pulses, LIDAR creates point clouds that represent the spatial distribution of objects and surfaces in the environment. These point clouds are used for various purposes, including mapping, obstacle avoidance, and object recognition.

The standard deviation in a LIDAR laser scan provides insights into the dispersion of individual data points within the scan. In the context of industrial warehouses, a low standard deviation is desirable because it indicates that the measurements are closely clustered around the mean distance value. This high precision is crucial for tasks such as:

Obstacle Detection and Collision Avoidance: The TC200 operating in warehouses needs precise distance measurements to navigate safely and avoid collisions with obstacles. A low standard deviation ensures that the LIDAR data accurately represents the distances to objects, reducing the risk of collisions.

Path Planning: The TC200 also uses LIDAR data to plan its paths efficiently. Low standard deviation in laser scans leads to more accurate mapping of the environment, enabling precise path planning and optimization of logistics operations.

In order to validate the laser scan, the TC 200 was allowed to perform a scan in the respective environment. During this time, the laser scan is published to the "/merged_cloud" topic. However, this data is raw and is collected from the entire environment. This is unneeded and unusable. Hence, this data is then converted to point cloud data through Point Cloud Library (PCL) and is now in the form of x, y, and z coordinates through a point generator. Moreover, the scan is then limited to identifying just the wall component in the environment and ignoring all other objects. This is done by limiting the scan range of the x and y components. Any points landing between x of -1.5 and +1.5 and y of -1.2 and +1.2, are considered fallen on the wall. Hence only those points are then further processed. Furthermore, the points are then compared against the best fit line created by a RANSAC algorithm. Following the RANSAC regressor, the best-fitting line is fit against the point cloud of the wall. This is how the robot recognises a wall and travels parallel along it. Therefore, as previously mentioned, a certain few parameter can be used to understand how consistent and accurate this line is. For this, three parameters are used. Their importance as well as their interpretation for this specific validation is as follows:

1. **Mean (Mean Error):**

- Importance: The mean error represents the average discrepancy between the observed laser points and the points predicted by the RANSAC line. It's a measure of how well the line approximates the data on average.
- Interpretation: A lower mean error indicates that, on average, the laser points are closer to the RANSAC line. This suggests a better fit to the data.

2. **Root Mean Square Error (RMSE):**

- Importance: RMSE measures the square root of the average of the squared errors (residuals). It provides an overall assessment of how well the RANSAC line fits the laser points.
- Interpretation: A lower RMSE signifies that the laser points are, on average, closer to the RANSAC line. It's a more comprehensive metric than the mean because it considers the magnitude of errors.

3. **Standard Deviation:**

- Importance: Standard deviation measures the spread or dispersion of the residuals from the RANSAC line. It indicates how consistent or variable the errors are.
- Interpretation: A lower standard deviation suggests that the residuals are more tightly clustered around the RANSAC line, indicating a more consistent and accurate fit. A higher standard deviation indicates greater variability or scatter.

In overall terms, Mean provides insight into the overall bias or average error in the fit. RMSE offers a comprehensive assessment by considering errors' magnitude and direction. Standard Deviation gives information about the spread or consistency of errors.

The average values for the corresponding metrics are shown in the Table:

Parameters	**Drywall**	**Glossy Door**	**Metal Shelf**
Mean	0.000693	-0.0498	-0.0013
RMSE	0.004171	0.3497	0.0154
Standard Deviation	0.004129	0.3462	0.0153
Minimum Error	-0.0122	-1.0720	-0.0291
Maximum Error	0.0131	0.4426	0.0265

Table 3: Deviation parameters categorised by wall material

It is inferred that the RMSE and the Standard deviation for all the environments were always either the same or very close in value. This may be due to symmetrical residual distribution. This means the points falling on the wall are equally likely to be on either side of the RANSAC

line. If there is no significant directional bias in the errors, it means that the RANSAC line provides a good overall fit to the data, and errors in both positive and negative directions are distributed fairly evenly. This checks out considering these are good environments for LIDARs and laser scan output.

Moreover, from Table 3, it can be observed that the Drywall was the best environment for laser scan outputs. It is ideal due to its matt, non-reflective finish. On the other hand, the glossy door happens to be the worst of the three environments with the highest deviation. It is especially notable that the minimum error is very outlandish because it is very likely a reflected laser point from the glossy door. The metal shelf is also not as good as the drywall but acts as a suitable yet unideal middle environment. However, it appears the reflections are not an issue here, likely due to the matt finish of the shelves. This study shows the effectiveness of the Hukoyu LIDARs that are used by the TC200 and their vulnerability to the respective environments in which it is placed. Hence, the best results can be expected when the walls of the environment are smooth, non-reflective and matt-coated.

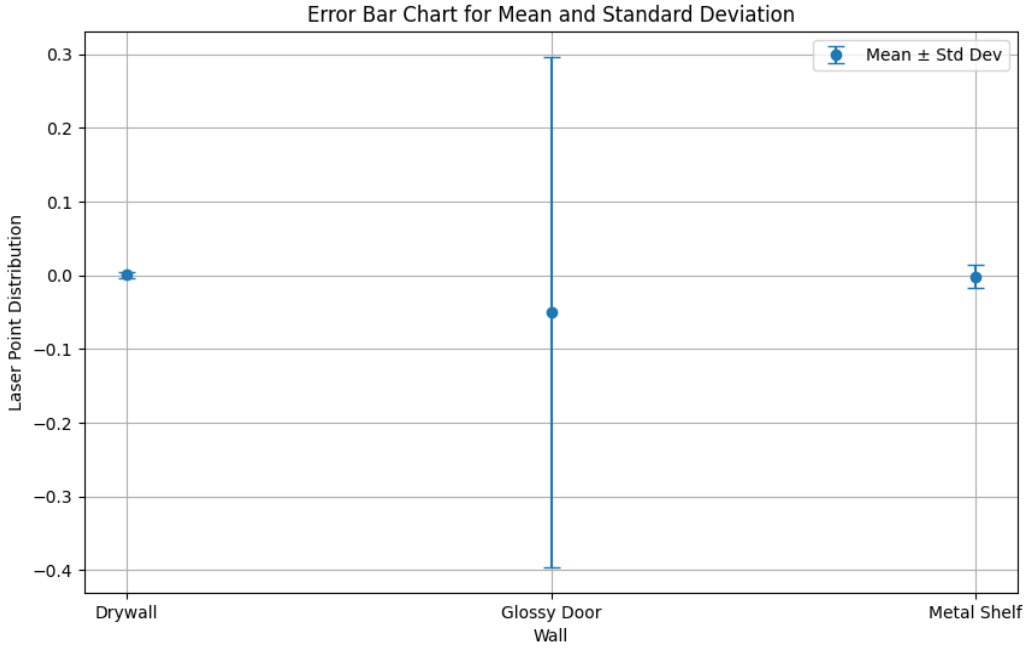

Fig. 17: Error Chart of Mean and Standard Deviation of Laser Points

5.1.2 Validation of P Controller

The validation of the Proportional controllers used in this specific use case implementation is done by tracking the trajectory of the TC200 robot while using the two P controllers that were elaborated in section 4.3.2 above. For this purpose, the robot was loaded in a warehouse environment and was then programmed to move parallel to a long shelf, holonomically by giving a Linear Velocity in the y-direction of TC200.

During this type of movement, the first P Controller came into action and the error caused by the disparity between the target distance of the robot from the wall, set by the user, and the intercept of the best-fit line arising from the RANSAC Algorithm fit line, was controlled by a Proportional Gain value (Kp_distance) of -0.5, which was multiplied by the distance error. This

value was given to the Linear Velocity of the robot in the X-direction, in order to control the robot's distance from the wall.

The second P Controller controlled the robot orientation (θ) where the difference between the target orientation of the robot, which is predefined as 0.0, and the slope arising from the best-fit line from the RANSAC Algorithm. This difference in the orientation angle of the robot is then controlled by multiplying the angle error by a Proportional Gain value (Kp_angle) of 0.5. This value was given to the Angular velocity of the robot in z-direction, in order to control the robot orientation.

These values of the Proportional Gains for both the P controllers were fine-tuned, so that the robot moves smoothly, without exhibiting noticeable oscillations during the robot movement.

- **For P Controller 1**:-

  Distance Error (delta_distance) = target distance - distance from the wall at that instance
  Kp_distance = - 0.5
  linear_vel_x = kp_distance * delta_distance

- **For P Controller 2**:-

  Angle Error (delta_angle) = target orientation angle - orientation angle at that instance
  Kp_angle = 0.5
  velocity_msg.angular.z = angular_vel

The plot below, Fig. 18, shows the trajectory the TC200 Autonomous Mobile Robot exhibits when moved holonomically along the shelf, in a warehouse environment. The combined effect of both the Proportional Controllers, controls the robot's movement to be fixed at a particular distance to the wall of the shelf and be in a desired orientation, such that the camera mounted on the robot, faces the wall of the shelf at every instance, in order to detect the desired objects, while the robot moves holonomically along the shelf.

The Actual Trajectory plot is generated by subscribing to the ‘odom’ topic of TC200 to get the positions in x and y along with the orientation of the robot and then plotting the positions as plot.plt and orientation as plot.scatter. The Ideal plot of TC200 is generated by subscribing to ‘/front/scan’ topic of LIDAR sensor, to get the ranges of laser scan according to the real world. Some small peaks in the Ideal Trajectory plot are due to some disturbances in the environment.

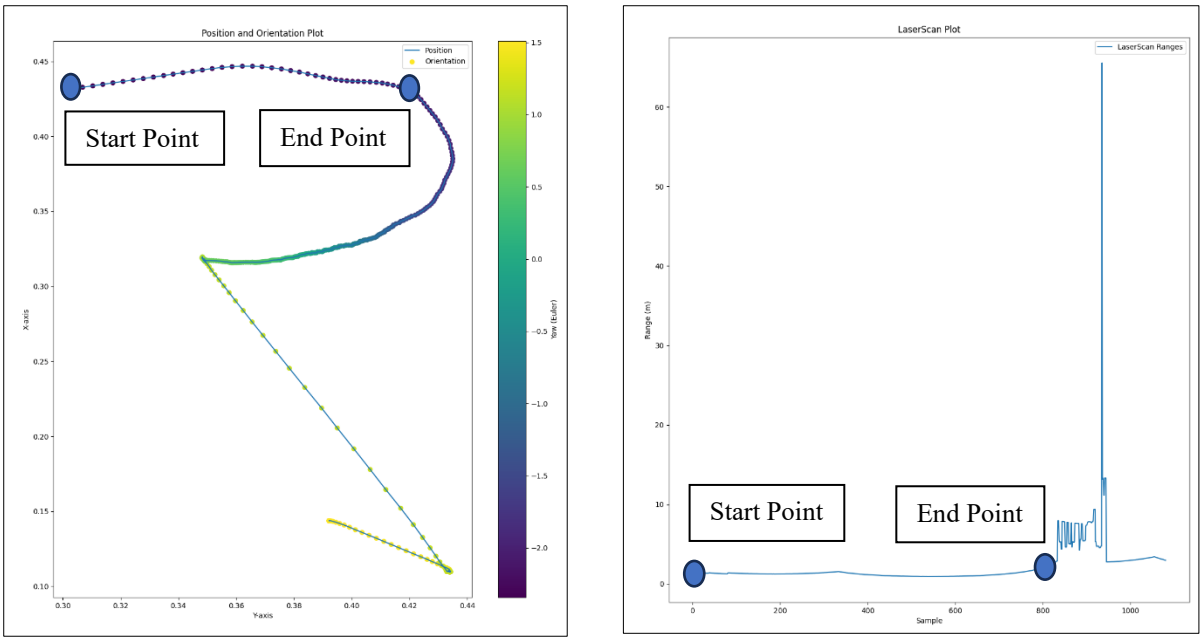

Fig. 18: Actual and Ideal Trajectory of TC200 during Use Case Implementation

From both the plots, it can be observed that the Actual Trajectory of TC200 while moving parallel to the shelf denoted by the straight laser scan data in the Ideal Trajectory plot, is not as expected. There is a slight deviation from the start to end point of the actual trajectory, when compared to the actual trajectory of TC200. Due to the disturbances in the environment, there are deviations in the LIDAR data that is used in the RANSAC algorithm. As the slope and intercept of the best-fit line are used in the P controllers, as mentioned above, we observe this actual trajectory instead of a smooth parallel line. But still, the P controllers do a good job in controlling the orientation angle as well as the distance between robot and target wall.

5.2 Computer Vision Module

5.2.1 Validation of Object Detection Model

To validate the object detection model, experiments with three training datasets were conducted, each comprising 100, 150, and 250 images. Consistent object instance ratios were maintained and included 15-20% background images in each dataset. Various augmentation techniques were applied during training to assess their impact on performance.

Validation Dataset	Number of Images	Red Bin Instances	Yellow Bin Instances
Validation	35	36	38

Test Dataset	Number of Images	Red Bin Instances	Yellow Bin Instances
Test	15	22	26

Table 4: Validation and test data set

To interpret the results, please refer to the following explanation of the validation parameters:

- 1) and 2) rows represent the percentage of correct detections for red and yellow bin instances in the test dataset.
- 4) and 5) rows indicate the percentage of correct detections for red and yellow bin instances in the validation dataset.
- 3) row signifies instances where the background was mistakenly identified as Red/Yellow on the test dataset.
- 6) row denotes instances where the background was erroneously identified as Red/Yellow on the validation dataset.

Sr. No.	Validation Parameters	Model Trained on						
		100 images	100 images with augmentation (copy-paste & hue)	150 images	150 images with augmentation (copy-paste & hue)	250 images	250 images with augmentation (copy-paste & hue)	250 images with augmentation (copy-paste)
1	Red Bin Instances for test dataset	21/22 = 95.45 %	22/22 = 100 %	21/22 = 95.45 %	20/22 = 90.90 %	22/22 = 100 %	21/22 = 95.45 %	22/22 = 100 %
2	Yellow Bin Instance for test dataset	24/26 = 92.30 %	24/26 = 92.30 %	24/26 = 92.30 %	26/26 = 100 %	26/26 = 100 %	24/26 = 92.30 %	26/26 = 100 %
3	False Background instances for test dataset (instances)	2	3	2	1	1	1	0
4	True positive Detection of Red Bin Instances for validation dataset	35/36 = 97.22 %	35/36 = 97.22 %	36/36 = 100 %	36/36 = 100 %	36/36 = 100 %	36/36 = 100 %	36/36 = 100 %
5	True positive Detection of Yellow Bin Instance for validation dataset	37/38 = 97.36 %	38/38 = 100 %	38/38 = 100 %	38/38 = 100 %	38/38 = 100 %	38/38 = 100 %	38/38 = 100 %
6	False Background instances for validation dataset (instances)	13	12	14	11	5	5	5

Table 5: Object detection model training validation on different number of images

As the size of the training dataset was expanded, a notable improvement in true positive object detection rates, rising from approximately 92-97% to a perfect 100% for both the test and validation datasets, was observed. Concurrently, the rate of false background detections diminishes with the increasing number of training images.

In the final two columns of the analysis, a comparison of the results obtained using different augmentation techniques was conducted. It became evident that in comparison to combining copy-paste and hue augmentation, employing only copy-paste augmentation yields favourable outcomes.

5.2.2 Validation of SAM Model Annotations

1. **Validation on Test Dataset**

Since the Segment Anything model auto annotates the image for mask labels, its result must be validated to ensure that the correct labels are sent for training the Yolov8 Instance segmentation model.

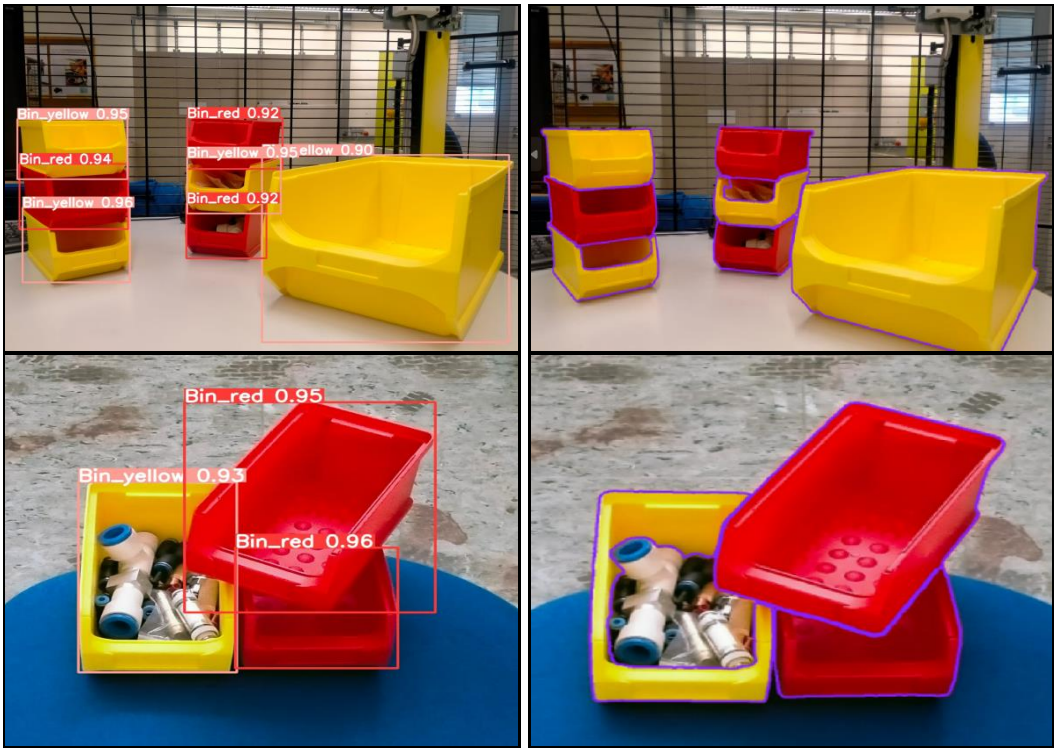

Fig. 19: Result of SAM auto annotation on boundary box input

SAM's auto annotation proves highly effective when the Object Detection (OD) model provides accurate labels, especially in complex images with numerous objects and intricate arrangements. This success highlights SAM's potential as a valuable tool for segment label annotation. However, there is still room for improvement in the OD model to enhance the accuracy of generated boundary boxes.

2. **Validation in Real Time on TC200**

Despite being slow for real-time implementation with a frame rate of 10 - 12 fps, The SAM model does accurate annotation and creates a well-defined boundary/mask

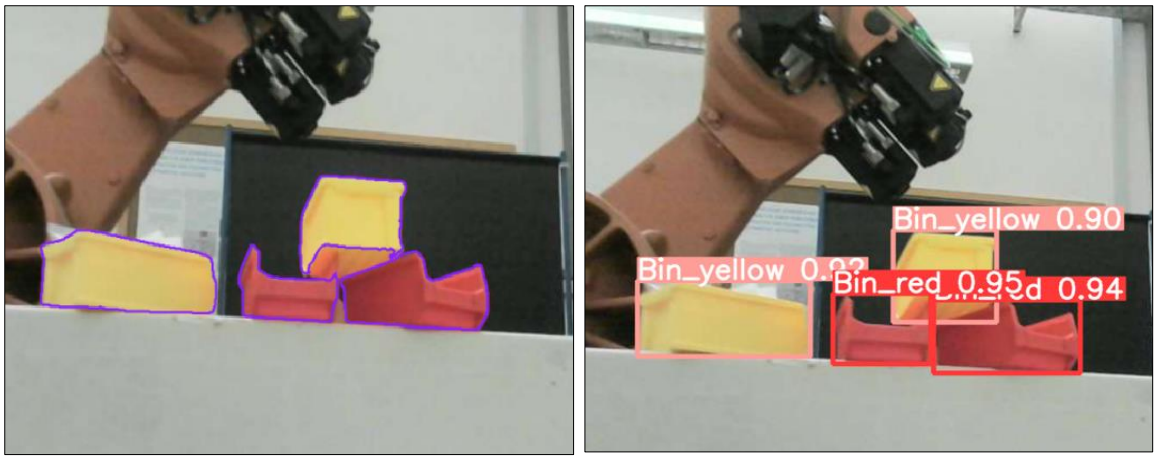

Fig. 20: Validation of Auto Annotation from SAM in real time

3. Outliers in SAM model Annotation

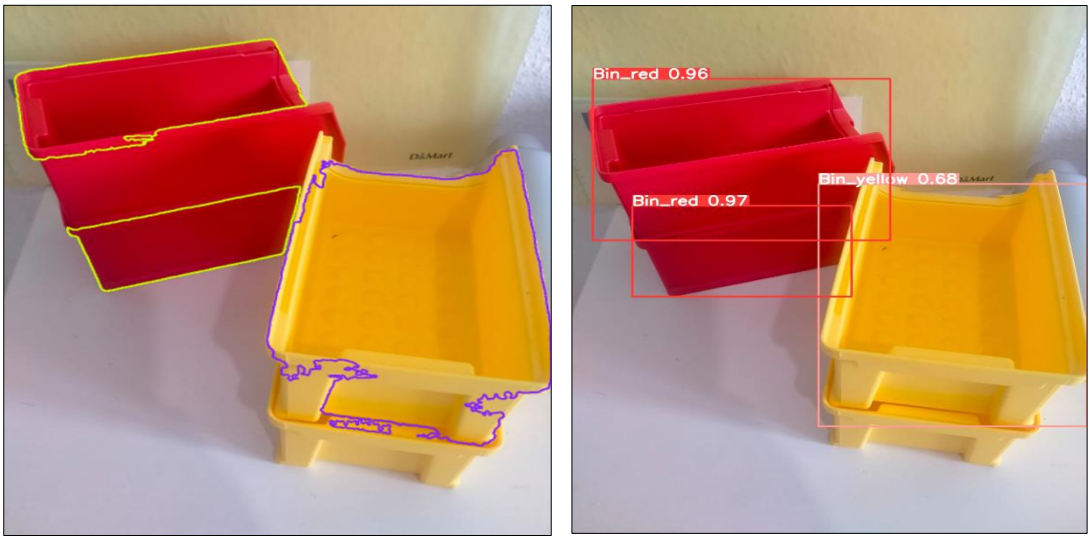

Fig. 21: Outliers in Auto Annotation by SAM model

Occasionally, SAM encounters challenges in producing perfect segment annotations around objects when the Object Detection (OD) model's annotations are subpar or there is a substantial variation in brightness and contrast on the object due to lighting conditions and shadow. This issue becomes evident when examining certain sample images as references. After a rough estimation, it appears that this type of problematic image accounts for approximately 2-3 percent of the entire dataset.

5.2.3 Validation of the Instance Segmentation Model

In this validation process, rigorous evaluation of the model's ability to correctly segment objects of interest from their backgrounds, determining its accuracy, robustness, and generalisation capabilities was done. By meticulously examining a range of parameters, metrics, and visualisation techniques, valuable insights into the model's strengths and

weaknesses were gained. In this section, a comprehensive analysis of the YOLO segmentation model's validation is presented.

Confusion matrix:

A confusion matrix is a visualisation that shows how your model is performing in the classes in which it was trained. This validation was carried out on 140 images. The rows of the confusion matrix represent the true labels of the objects in the dataset, while the columns represent the predicted labels of the objects in the dataset. The off-diagonal elements of the confusion matrix represent the number of objects that were misclassified. The diagonal elements of the confusion matrix represent the number of objects that were correctly classified. The confusion matrix in Fig. 21 shows that the instance segmentation of the object on the validation data set is very good, there are very few false detections.

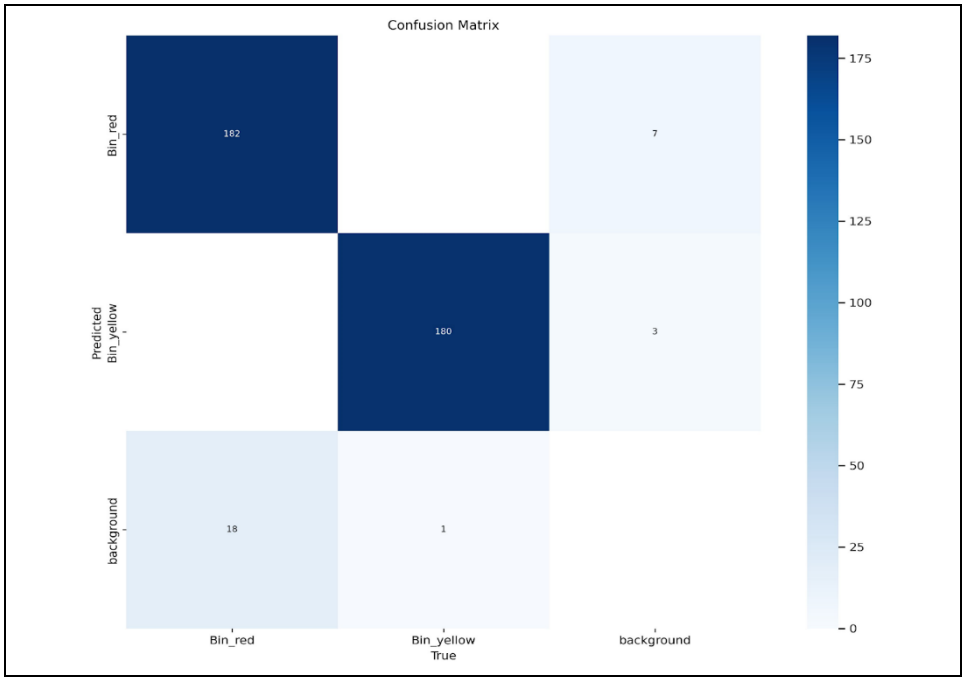

Fig. 22: Confusion matrix of Instance Segmentation Yolov8 custom model

Precision-Recall Curve:

A precision-recall curve to assess the trade-off between precision and recall at different confidence thresholds. A good model should have a curve that approaches the upper-right corner.

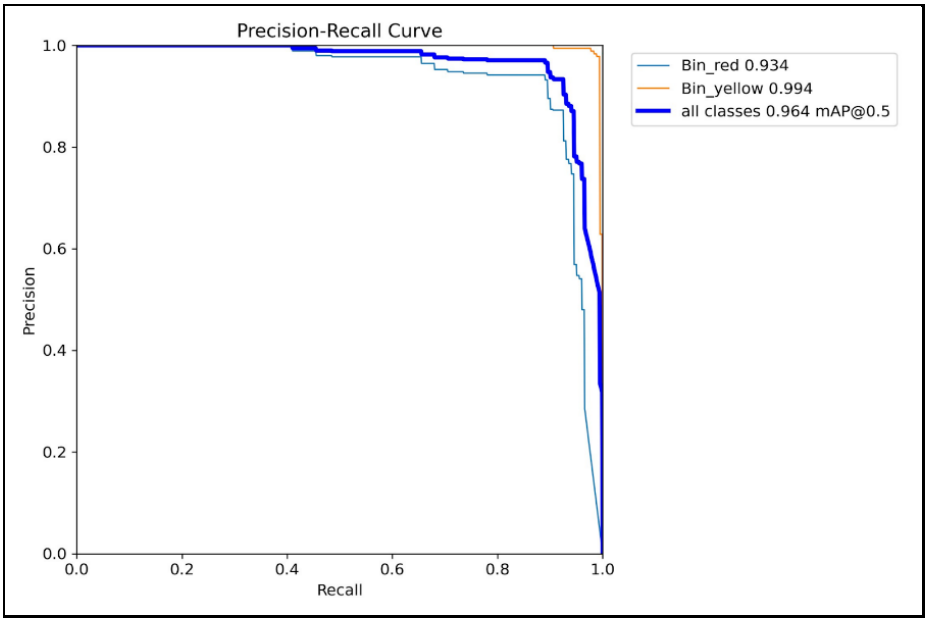

Fig. 23: Precision Recall graph of training the Instance Segmentation model

Loss Curves:

The loss curve shows how the model's loss changes over time during training and validation. Initially, the loss may be high as the model starts with random weights, but it should decrease as the model learns from the data.

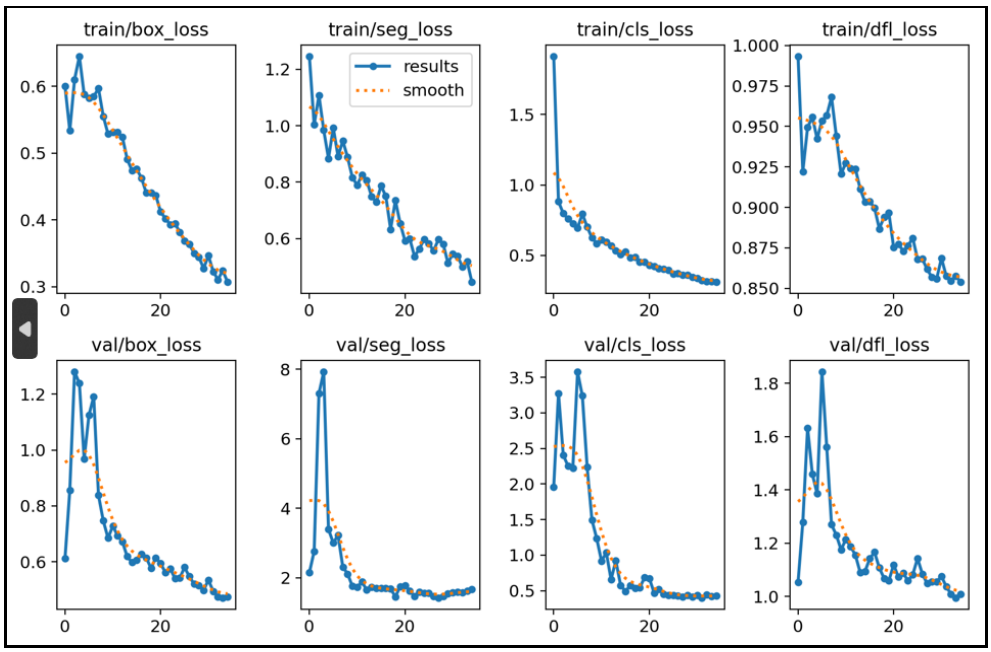

Fig. 24: Various Loss curves of training the Instance Segmentation model

Validation of Instance Segmentation model on Test Images:

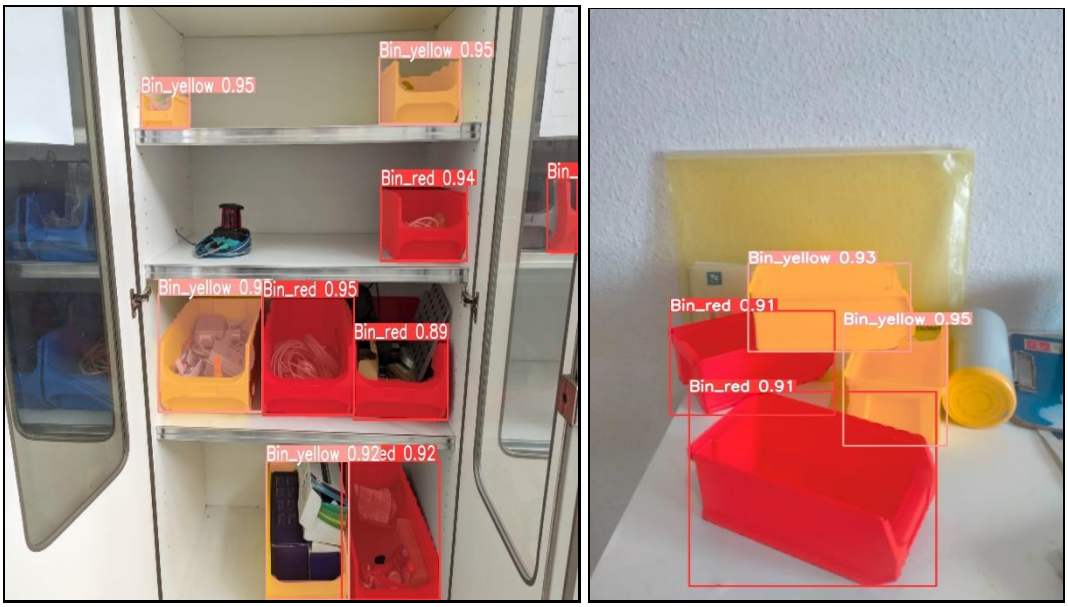

Fig. 25: Validation of instance segmentation on test images

Result:

Test Dataset	30 images	
Red_bin instances	45 instances	45 instances detected
Yellow_bin instances	47 instances	47 instances detected

Table 6: Validation of instance segmentation on test images

The test dataset comprised both intricate scenarios featuring multiple stacked bins in a new environment and straightforward images. Remarkably, the instance segmentation model consistently and accurately identified all the bins without any false detections when the confidence threshold was set at 60%. This level of performance demonstrates the model's robustness.

Validation of Instance Segmentation Modelon Videos

The model worked well with video with videos shot from a mobile phone, the percentage of false detection was roughly 2-3% and around 1% of frames had no detection even though the object was present.

This was validated by taking multiple videos and then running it on the instance segmentation model, The number of total frames was counted and the number of detection frames was counted. The Frames per second were around 10-12 fps, when the model detects objects it drops to 7-8 fps. due to processing.

5.2.4 Validation of SORT Tracker in Use Case

To validate the performance of the tracker the model was run on the robot in real time. The robot was run at different speeds and different numbers and sizes of the bins were detected to evaluate the robustness of the tracker. The readings were taken over 5 cycles for each speed.

Robot Speed	1 large bin	1 small bin	3 bins	5 bins
0.04 m/s	100% detection	100% detection	100% detection	100% detection
0.06 m/s	100% detection	100% detection	100% detection	92% detection
0.07 m/s	80% detection	60% detection	73% detection	64% (detection)
0.08 m/s	60% detection	40% detection	33% detection	36% (detection)

Table 7: Validation of instance segmentation model on the robot in Real time

The findings indicate that as the speed of the robot escalates, the tracker's capacity to identify and stop the robot for the object reduces gradually. Optimal detection is achieved at speeds of up to 0.06m/sec. This phenomenon can be attributed to factors like the number of frames, lighting conditions, and field of view, among others.

Chapter 6: Conclusion

In conclusion, this project represents a significant leap forward in the realm of warehouse management systems, offering a holistic solution to address the challenges faced by traditional setups. The project's multifaceted approach has encompassed several critical components, each contributing to its overall success.

First and foremost, the utilisation of the TC200 Autonomous Mobile Robot (AMR) brings autonomous navigation to the forefront. This enables the robot to autonomously traverse the warehouse, deftly avoiding obstacles while optimising path planning. By automating navigation, this project eliminates the need for manual laborious tasks and reduces the risk of human errors, significantly enhancing warehouse productivity.

Object tracking, the project's second key component, leverages advanced computer vision algorithms and sensors like a camera and LIDAR to precisely detect and monitor objects, particularly coloured storage bins. This innovation streamlines the retrieval process, making it faster and more efficient. It also minimises errors in object identification, contributing to the project's overall accuracy.

Moreover, the project's scalability factor ensures that it remains adaptable to ever-evolving warehouse layouts and increasing demands. This flexibility positions it as a future-ready solution, capable of evolving in tandem with the dynamic nature of modern warehouses.

Safety, an essential aspect of any industrial setting, is paramount in this project. The TC200 AMR incorporates cutting-edge safety features, including obstacle detection and collision avoidance systems. This emphasis on safety ensures seamless collaboration between human workers and autonomous robots, minimising the risk of accidents and injuries.

The current implementation of the use case extends to future possibilities and the scope of the overall project. The introduction of a Robotic Manipulator Arm, mounted on the top of the TC200 AMR, enables the picking and placing capabilities for the mobile robot, which is the natural extension to the achieved solution, for the use case. Also, the mounted camera opens the possibilities for endless advancements, beyond the scope of the project. Sophisticated Computer vision models like the Segment Anything Model can be clubbed with natural language processing. This innovation eliminates the need for an expanded image dataset, as user-provided text or speech prompts can effectively drive the instance segmentation process. This forward-thinking approach promises to elevate the project's potential impact and relevance in industrial automation

All in all, the project's contribution to the industry is substantial. It exemplifies the potential of advanced automation and robotics in revolutionising warehouse management systems. By reducing reliance on manual labour, the project optimises productivity, reduces errors, and enhances overall efficiency. It sets a benchmark for future endeavours in the field, emphasising the importance of flexible and adaptive automation solutions in modern warehousing.

Citations

[1] Nuno Correia, Leonor Teixeira, Ana Luísa Ramos, “Implementing an AGV system to transport finished goods to the Warehouse”, March 2020. Institute of Electronics and Informatics Engineering of Aveiro / Research Unit on Governance, Competitiveness and Public Policies, Department of Economics, Management, Industrial Engineering and Tourism, University of Aveiro, 3010-193 Aveiro, Portugal.

[2] Giuseppe Fragapane, Renéde Koster, Fabio Sgarbossa, Jan Ola Strandhagen, “Planning and control of autonomous mobile robots for intralogistics”, January 2021. Department of Mechanical and Industrial Engineering and Norwegian University of Science and Technology.

[3] Lu ZHEN, Haolin LI. “A literature review of Smart Warehouse Operations Management”, June 2021. School of Management, Shanghai University, Shanghai 200444, China.

[4] Niall O’ Mahony, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan, Joseph Walsh,“Deep Learning vs. Traditional Computer Vision”, IMaR Technology Gateway, Institute of Technology Tralee, Tralee, Ireland niall.omahony@research.ittralee.ie

[5] Jérôme Rutinowski, Hazem Youssef, Anas Gouda , Christopher Reining,Moritz Roidl, “The Potential of Deep Learning based Computer Vision in Warehousing Logistics”, Chair of Material Handling and Warehousing Faculty of Mechanical Engineering TU Dortmund University

[6] Alexander Naumann, Felix Hertlein, Laura D, Steffen Thoma, Kai Furmans, “Literature Review: Computer Vision Applications in Transportation Logistics and Warehousing”,FZI Research Center for Information Technology, Karlsruhe Institute of Technology (KIT).

[7] “Supported Modes” https://docs.ultralytics.com/models/yolov8/#supported-modes

[8] “What are the differences between AGV and AMR robotic solutions?” - https://inviarobotics.com/blog/key-differences-between-amr-and-agv-robots/

[9] “AGV vs AMR:Differences and Advantages-KNAPP” - https://www.knapp.com/en/insights/blog/differences-between-agv-amr/

[10] “TC200 - Génération Robots.” *TECDROM TC200 Mobile Robot*, www.generationrobots.com/media/tecdron/flyer-TECDRON-TC200-Mobile-Robot-ros.pdf.

[11] Alharbi, M.; Karimi, H.A. A Global Path Planner for Safe Navigation of Autonomous Vehicles in Uncertain Environments. Sensors 2020, 20, 6103. https://doi.org/10.3390/s20216103

[12] R. Guldenring, M. Görner, N. Hendrich, N. J. Jacobsen and J. Zhang, "Learning Local Planners for Human-aware Navigation in Indoor Environments," 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 2020, pp. 6053-6060, doi: 10.1109/IROS45743.2020.9341783.

[13] Apurin, A., Abbyasov, B., Martínez-García, E.A., Magid, E. (2023). Comparison of ROS Local Planners for a Holonomic Robot in Gazebo Simulator. In: Ronzhin, A., Sadigov, A., Meshcheryakov, R. (eds) Interactive Collaborative Robotics. ICR 2023. Lecture Notes in Computer Science(), vol 14214. Springer, Cham. https://doi.org/10.1007/978-3-031-43111-1_11

[14] "base_local_planner - ROS Wiki." http://wiki.ros.org/base_local_planner

[15] Looi, Chen Zheng, and Danny Wee Kiat Ng. "A study on the effect of parameters for ROS motion planer and navigation system for indoor robot." International Journal of Electrical and Computer Engineering Research 1.1 (2021): 29-36.

[16] "navfn - ROS Wiki." http://wiki.ros.org/navfn

[17] "MoveBase" move_base - ROS Wiki

[18] "Wiki." *Ros.Org*, wiki.ros.org/ROS/Tutorials/MultipleMachines

[19] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D. Cubuk, Quoc V. Le1 Barret Zoph, "Simple Copy-Paste is a Strong Data Augmentation Method for Instance Segmentation", Google Research, Brain Team, UC Berkeley, Cornell University

[20] "YOLO models for Object Detection Explained [YOLOv8 Updated]" https://encord.com/blog/yolo-object-detection-guide/

[21] "Instance Segmentation Vs. Object Detection: 3 Things You Need To Know" https://www.folio3.ai/blog/instance-segmentation-vs-object-detection/

[22] "Robust Instance Segmentation through Reasoning about Multi-Object Occlusion" https://openaccess.thecvf.com/content/CVPR2021/papers/Yuan_Robust_Instance_Segmentation_Through_Reasoning_About_Multi-Object_Occlusion_CVPR_2021_paper.pdf

[23] "Semantic Segmentation vs. Instance Segmentation Explained" https://blog.roboflow.com/difference-semantic-segmentation-instance-segmentation/

Appendix

A 1.1 Comparison of popular Manual annotation tools

Feature	LabelMe	LabelImg	VGG Image Annotator (VIA)
Platform	Web-based	Desktop	Web-based and Desktop
Open Source	Yes	Yes	Yes
Annotation Types	Polygon, Rectangle	Rectangle, Polygon	Polygon, Rectangle, Point
Export Formats	JSON, XML, Others	PASCAL VOC, YOLO	JSON, CSV, Others
Customizable Labels & Multi class support	Yes	Yes	Yes
Offline Availability	No	Yes	Yes
GitHub Maintained	Yes	Yes	Yes
Community Support	Active	Active	Active
Documentation	Available	Available	Available
User Interface	Web-based UI	GUI	Web-based UI and Desktop UI
Ease of Use	Moderate	Easy	Moderate
Updates and Maintenance	Varies	Varies	Active

A 1.2 Comparison of popular Supervised annotators:

Criteria	CVAT (Computer Vision Annotation Tool)	Hasty.ai	YAT	AutoML Vision Edge
Annotation Types	Bounding Boxes, Polygons, Key Points, Lines, etc.	Bounding Boxes, Polygons, Points, Lines, etc.	Bounding Boxes, Polygons, Lines, Points, etc.	Various
Ease of Use	Intuitive and feature-rich	User-friendly with AI assistance	Moderate	Moderate
Automation	Manual, but supports semi-automation	AI-assisted annotation	AI-assisted	Yes
Auto Annotation	Not available	Available	Available	Available
SAM Support	Limited	Limited	Limited	Yes
Export Formats	Common formats (COCO, Pascal VOC, etc.)	Common formats (COCO, VOC, etc.)	Common formats (COCO, Pascal VOC, etc.)	Common formats (COCO, Pascal VOC, etc.)
Cost	Free and open-source	Pricing available upon request	Pricing available upon request	Pricing available upon request
Platform	Web-based (self-hosted option)	Web-based	Web-based	Web-based
GitHub Maintained	Yes	Yes	Yes	Not specified
Scalability	Suitable for small to medium datasets	Supports large datasets	Supports large datasets	Suitable for small to medium datasets
Community and Support	Active user community and documentation	Availability of support	Availability of support	Limited documentation